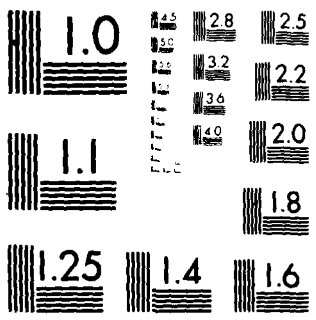

1.0
4.5
5.0
5.6
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

(12)

Systems
Optimization
Laboratory

Department of Operations Research
Stanford University
Stanford, CA 94305

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER SOL 81-21	2. GOVT ACCESSION NO. AD-A110845	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Polynomial Local Improvement Algorithms in Combinatorial Optimization		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Craig A. Tovey		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0267 DAAG29-81-K-0156
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Operations Research - SOL Stanford University Stanford, CA 94305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR-047-143
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research - Dept. of the Navy 800 N. Quincy Street Arlington, VA 22217		12. REPORT DATE November 1981 13. NUMBER OF PAGES 63
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) U.S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release and sale;
its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

THE VIEW, OPINIONS, AND/OR FINDINGS CONTAINED IN THIS REPORT
ARE THOSE [illegible] CONSTRUED AS
AN OFFICIAL [illegible] OR DE-
CISION, UNLESS SO DESIGNATED BY OTHER DOCUMENTATION.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

optimization	local improvement algorithm
optimal adjaceny algorithm	worse case behavior
expected iterations	linear programming
integer programming	complementary pivoting

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

SEE ATTACHED ABSTRACT

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

SOL 81-21: Polynomial Local Improvement Algorithms in Combinatorial Optimization, Craig A. Tovey

Local improvement algorithms are widely used to solve a variety of problems, including linear programming, integer programming, and linear complementarity. One reason for the popularity of local improvement methods is that they tend to run quickly for problems encountered in practice. On the other hand, for some particular optimization problems, worst case examples have been constructed whose performance grows exponentially with the size of the problem. It is natural to ask whether these are characteristics of local improvement algorithms in general, and to seek a more exact description of the performance of these algorithms.

The subject of this report is an analysis of the expected, or average case performance of local improvement algorithms. The first chapter presents the basic model, defines the combinatorial structures which are the basis for the analysis, and describes the randomness assumptions upon which the expectation are based. The second chapter examines these structures in more detail, including an analysis of both best and worst case performance. The third chapter discusses simulation results which predict an approximately linear average case performance, and proves an $O(n^2 \log n)$ upper bound for two of the random distributions assumed. Chapter Four proves some extensions and sharper versions of this upper bound. The fifth chapter applies the model to principal pivoting algorithms for the linear complementarity problem, and to the simplex method. Although local improvement is not guaranteed to find a global optimum for all problems, most notably those that are NP-complete, it is nonetheless often used in these cases. Chapter Six discusses these applications.

SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA 94305

POLYNOMIAL LOCAL IMPROVEMENT ALGORITHMS IN COMBINATORIAL OPTIMIZATION

by

Craig Aaron Tovey

TECHNICAL REPORT SOL 81-21

November 1981

Research and reproduction of this report were partially supported by the Department of Energy Contract AM03-76SF00326, PA# DE-AT03-76ER72018; Office of Naval Research Contract N00014-75-C-0267; National Science Foundation Grants MCS76-81259, MCS-7926009 and ECS-8012974; and Army Research Office Contract DAA29-79-C-0110.

TABLE OF CONTENTS

Chapter Page

CHAPTER 1.
A MODEL OF LOCAL IMPROVEMENT ALGORITHMS

1.1 Introduction

Suppose you are at the base of a mountain in the middle of the night, and you want to climb to the top. The moon is dim, so all you have to to light the way is a flashlight, which lets you see the ground within a ten foot radius. What do you do? You sweep the area around you with the flashlight, find a nearby spot that is a little higher than where you are standing, and move there. Then you repeat this process, always increasing your elevation, until you reach a peak— a point higher than the area around it. If the mountain has only one peak, you are guaranteed to get to the top. Such a method is called a *hill climbing* or *local improvement* algorithm.

Hill climbing methods are used in algorithms for linear programming, linear complementarity, artificial intelligence applications, and many other combinatorial optimization problems. Most of the analysis of these methods has dealt with the worst case performance of a particular algorithm. More recently, there has been some work concerning the average case performance of the Simplex Method (Ross, 1981) which, however, fails to take into account the combinatorial structure of the problem. We present a general model which overcomes this difficulty and which can be applied to a variety of local improvement algorithms.

In an optimization problem, the horizontal cross section of the mountain corresponds to the domain, or search space; the height of a spot of ground corresponds to the value of the objective function we wish to maximize over the domain. Suppose, then, we have a real valued function f whose domain is the set of vertices of the n-cube (i.e., the set of n-tuples of zeroes and ones), and that we wish to maximize $f(x)$ over this space. We can assume that all the values of f are distinct, for if $f(x) = f(y)$ we say $f(x) > f(y)$ if x is lexicographically greater (see Dantzig, 1963) than y. For example, if $f(0110) = f(0101)$ we say that $f(0110)$ is the larger.

The domain of the function can be thought of as a set of boolean decision variables: many optimization problems may be cast in this form (see Cook, 1971). If f is to be minimized, we maximize $-f$. In some applications such as the simplex method, not all vertices of the hypercube correspond to feasible points. If a vertex x is not feasible, we apply penalties for constraint violation to the value of $f(x)$.

There is a natural notion of distance between two vertices of the n-cube: the number of components in which they differ. This distance is a metric and is known as the Hamming distance (it equals the square of the Euclidean norm). If x and y are at a distance of zero, then $x = y$; if x and y are at a distance of one, they share an edge and are said to be *adjacent* or *neighbors*. A vertex whose function value is greater than any of its n neighbors is called a local maximum. If f has the property that a local maximum is a global maximum we say that f is *Local-Global* or *LG* for short. The LG property is of course reminiscent of the property that a local minimum of a convex function is a global one; see also (Dearing, 1976).

If f is LG, a local improvement algorithm will solve the problem of maximizing f over the stated domain. In particular, we define the Optimal Adjacency (*OA*) algorithm as follows:

Optimal Adjacency Algorithm

```
0. Start with any vertex $x$.
1. If $x$ is locally optimal, stop with $x$ the solution. Otherwise proceed to 2.
2. Let $y$ be the optimal vertex adjacent to $x$. Set $x$ equal to $y$ and go to 1.
```

Since the domain is finite and has only one local optimum, the algorithm must terminate after finitely many steps with the correct solution.

Note that every vertex has only n neighbors and that the diameter of the space (the maximum possible Hamming distance between any two vertices) is also n. A single iteration of the OA algorithm requires at most n function evaluations, and it is quite possible for the number of iterations to be a low order polynomial. The next section deals with the problem of how many iterations the algorithm can be expected to take for a particular instance of an LG problem.

1.2 OATs.

If we are given a particular LG function f, we can construct a directed tree to show how many iterations the optimal adjacency algorithm will require:

i) Each vertex of the n-cube corresponds to a node of the tree.
ii) The father of a vertex is its optimal adjacent vertex; if a vertex is a local optimum, it has no father.

The tree is called an Optimal Adjacency Tree, or OAT. Its root is the local (hence globally unique) optimum. The OAT displays the path followed by the algorithm by going from son to father on the tree (a biologically backwards progression.) It should be emphasized that for any instance of any local-global problem there is a unique OAT which describes the action of the OA algorithm on that instance. Figure 1.1 shows four possible OATs for n=2,

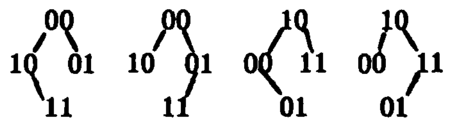


Figure 1.1

and Figure 1.2 shows a possible OAT for $n = 3$.

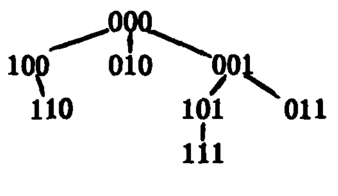


Figure 1.2

The set of OATs of order n has a 2^n-fold symmetry. Any of the 2^n vertices could be the root; from here on we always assume for notational convenience that the origin, 00...0, is the root.

Suppose that we were going to run the OA algorithm on a particular instance of a problem, and we knew that this instance had the structure shown in Figure 1.2. How many iterations would we expect the algorithm to take? If the starting vertex is chosen at random, there would be an equal probability of starting at each of the 8 vertices. In general for each starting vertex, the path to the root in the OAT is by definition the path the optimal adjacency algorithm will follow. Thus the height or pathlength of each vertex in the tree is the number of iterations the algorithm would need to reach the optimum from that vertex. The *mean pathlength* of the tree (the mean of the pathlengths of all the nodes in the tree) is precisely equal to the expected number of iterations of the OA algorithm. Thus for a problem which has the structure of the tree in Figure 1.2, the OA algorithm would be expected to take $(1 \times 1 + 3 \times 2 + 3 \times 3 + 1 \times 4)/8 = 2\frac{1}{2}$ iterations. As another example, the expected number of iterations required for any LG problem with $n = 2$ is two, because all OATs for $n = 2$ have a mean pathlength of two.

If f is not local-global, the rules for producing the OAT will instead produce an OAF, or Optimal Adjacency Forest, with one tree per local optimum.

There exist classes of OATs which are exponentially high, which implies that the mean pathlength is also. (Note: the *height* of a tree is the maximum height of the vertices in the tree. A class of trees is *exponentially* high if the heights of the trees in the class grow exponentially in n.) The study of worst case OATs is closely related to the study of snakes-in-boxes; results from the latter can be applied to the former with little change. In the next chapter we show that a lower bound on the maximum height of an OAT of order n is given by

$$\frac{7 \times 2^n}{4(n-1)} - 1$$

for $n \geq 6$. This comes from a lower bound on snakes-in-boxes due to Victor Klee (Klee, 1970).

Since there exist OATs that are exponentially high, any strict bound on the performance of the OA algorithm must be exponential in n. If the worst case OAT is in some sense rarely encountered, it is of interest to ask, what is the expected performance of the optimal adjacency algorithm? Or, equivalently, what is the *expected* mean pathlength of an OAT of order n? This question is of course not well defined without a notion of an underlying probability distribution of OATs. There is strong empirical evidence that under a variety of underlying distributions, the expected mean pathlength of an OAT is linear in n. The next section defines the necessary terminology for describing these underlying probability distributions.

1.3 LG Orderings and the Boundary

When we construct an OAT from the function f, we are not interested in the specific numeric values of f, but in the ordering of values of f on the vertices. Since functional values are distinct, the vertices can be uniquely ordered from high to low function value. Such a list of vertices is called an ordering. For our purposes, the ordering of the vertices *defines* f. If f were as in Example A,

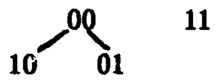
```
000
001
100
010        Example A
101
111
110
011
```

then the rules for OATs would produce the OAT we looked at in Figure 1.2. In this case we say the ordering *produces* the OAT.

Not every ordering produces an OAT; many produce OAFs instead. For instance, Example B,

```
00
11         Example B
01
10,
```

is an ordering for $n = 2$ which produces the OAF



because 00 and 11 are not adjacent.

If an ordering produces an OAT it is said to be an LG ordering. An obvious necessary and sufficient condition that an ordering be LG is that every vertex except the first have an adjacent vertex that is located higher up in the ordering. Example B fails to be LG because all of the neighbors of the vertex 11 are located below it in the ordering.

Each LG ordering produces only one OAT, but given an OAT, there can exist many orderings which produce it. For example, if the bottom two vertices in Example A traded places in the ordering, the resulting OAT would still be Figure 1.2.

How can we test an ordering for LG? We verify that each of its vertices except the top one has at least one neighbor above it. How can we produce an ordering that is local-global? One method would be to generate orderings at random until one is reached that passes the above test and is therefore LG. This method, though simple, has the disadvantage that the proportion of orderings that are LG becomes vanishingly small as n increases. There is a better method that produces only orderings. However, before presenting it, we need one more concept: the boundary.

Suppose we had an LG ordering written down somewhere, and all we could remember were the first five vertices. What could we deduce about the sixth? The sixth vertex would have to be adjacent to one or more of the first five, and distinct from them. If S is a subset of the vertices of the n-cube, we define the *boundary* of S to consist of all vertices x such that x is not in S and such that for some y in S, x and y are adjacent. It is obvious that the sixth vertex in any LG ordering must belong to the boundary of the first five; in general the $i + 1$st vertex in an LG ordering must be an element of the boundary for the first i vertices. To recursively enumerate all LG orderings, then, we use the following procedure.

Procedure Enumerate *(When this procedure is called it is passed the values of n, i, and the first i values of an array A[1,...,n] of vertices.)*

```
0. Begin
1. If $i=n$, output A and go to 5.
2. Compute the boundary, B, of $A[1], \ldots, A[i]$
3. Set $i := i+1$.
4. For each member $x$ of B, Do
    Begin
    Set $A[i] := x$
    Call procedure enumerate
    End
5. End
```

If we want to produce one LG ordering randomly, we change step 4 to "For some member x of B". Then at each step of the random generation process, a vertex x is selected from the boundary, assigned a father, and attached to the tree.

How we select x determines what the underlying distribution is. We could (theoretically speaking) choose x so as to give each LG ordering an equal chance of being produced: this distribution is called "all LG orderings equally likely." We could let each member of the boundary have an equal chance of being chosen: this distribution is called "boundary members equally likely" or just the "boundary" distribution for short. It can be thought of in the following way: if we know what the i best points are (best in the sense of best objective function value), then if the function is going to be LG, the $i+1$st best point must be a member of the boundary set of these i points; saying that these boundary points all have equal probability of being that $i+1$st best point is exactly what is meant by "boundary members equally likely."

Note: "boundary members equally likely" is not the same as "all LG orderings equally likely," because the size of the boundary at stage i varies depending on what the previous choices have been. Thus, an ordering that has an unusually large boundary in the early stages would be more likely to occur under the LG-orderings-equally-likely distribution than under the boundary distribution.

Every boundary member has at least one neighbor that has already been selected, but some have more such neighbors than others (n at most, since each vertex has n neighbors). An alternate criterion would be to give each boundary member a weight proportional to the number of chosen neighbors it has; thus vertices with more chosen neighbors are more likely to be chosen themselves. For reasons that will become clear later, this distribution is called the coboundary distribution. This notion of randomness may seem slightly preferable to the boundary distribution if one thinks that one can judge a vertex by its neighbors (not an unreasonable supposition for a local-global problem).

1.4 BATS

We have now defined three different possible distributions on local-global problems. From an ordering we can deduce the OAT which tells us exactly how many iterations the OA algorithm will take. Unfortunately, it is hard to analyze OATs theoretically because an OAT is so tightly constrained; what nodes can be attached in one place often depends on what nodes are atached somewhere else. For instance, the tree in Figure 1.3 is not an OAT:

$$\begin{tikzcd}
00 \arrow[d, no head] \\
01 \arrow[d, no head] \\
11 \arrow[d, no head] \\
10
\end{tikzcd}$$

Figure 1.3

The tree implies $f(11) < f(01) < f(00)$, so $f(11) < f(00)$. By the optimal adjacency rule, the father of 10 must be the member of the set $\{00, 11\}$ which has the larger function value. The above inequality implies that 00 should be the father, but in Figure 1.3 the father is 11, instead. This is a contradiction, so the tree is not an OAT. To make it an OAT, the node 10 would have to be moved and attached to the node 00. A more subtle example is shown below in Figure 1.4.

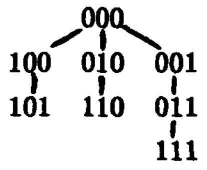


Figure 1.4

Consider the node 101: its three neighbors are 100, 001, and 111. By definition of the optimal adjacency rule, the one of these with largest function value must be the father of 101. Looking at Figure 1.4, we see that the father is 100. This implies that $f(100) > f(001)$. However, we can deduce by the same reasoning that $f(001) > f(010)$ and that $f(010) > f(100)$. Together these relations yield $f(100) > f(001) > f(010) > f(100)$, a contradiction. The tree in Figure 1.4 is therefore not an OAT. The problem is that the OA algorithm must choose the *best* neighbor to go to. Suppose we relaxed this condition, and only required that the algorithm proceed from a vertex to a *better* adjacent vertex. The trees in Figures 1.3 and 1.4 are consistent with the action of such an algorithm. Formally, we define the Better Adjacency (BA) algorithm as follows:

BA (Better Adjacency) Algorithm

1. Start at a random vertex x.

2. Search through x's neighbors until a better one, y, is found or all neighbors have been tried. In the former case set x equal to y and iterate (2.); in the latter case stop with x optimal.

This algorithm is valid for the same reasons as the OA algorithm; its representation is the Better Adjacency Tree, or BAT. Given any LG ordering, we can pick a higher valued neighbor for each vertex (except the origin) and make that neighbor its father. The resulting tree (it is a tree since it has no cycles and there is a path connecting all nodes to the origin) is a BAT.

For example, the tree in Figure 1.3 could have been generated from the ordering

00

01

11

10.

The tree in Figure 1.4 could have been generated from the ordering

000
100
010
001
101
110
011
111.

Of course, the set of BATs properly contains the set of OATs.

An OAT can be thought of as a directed graph $H = (V, F)$ whose vertex set V consists of the vertices of the n-cube, and whose directed edges consist of the son, father pairs in the OAT. Suppose we have an OAT and an ordering which produces it. If we assign to each vertex x a value π_x equal to its position in the ordering, then for every edge (x, y) in the graph, it must be true that $\pi_x > \pi_y$. This property of the π values is well known to be a necessary and sufficient condition that a directed graph be acyclic, i.e., that the graph has no directed cycles. Note that this property is necessary; if the graph H is acyclic we can derive an ordering that corresponds to it. The fact that we can "go the other way" and derive an ordering from the tree allows us to define OATs and BATs in graph theoretic terminology without explicit reference to function values. These definitions have the advantage of clarifying the distinction between OATs and BATs.

Let $G = (V, E)$ be a graph of the n-cube so that the nodes of G are the vertices of the n-cube, and there is an edge joining two vertices iff they are adjacent. Let $H = (V, F)$ be a directed subgraph of G with outdegree 1 (no node has more than one edge leaving it) and with $|F| = |V| - 1$ edges. Then H is a BAT iff it is acyclic. (The directed edges correspond to {son, father} pairs of the BAT.) Now define $H' = (V, F')$ so that F is contained in F' and for every directed edge (u, v) in F and edge (u, w) in G, the edge (w, v) is in F'. Note that H' is not a subgraph of G, because it contains these neighbor–to–father edges. Then H is an OAT iff H' is acyclic. The intuitive meaning is that when building BATs the only information in a {son, father} pair is that $f(son) < f(father)$; when building OATs, a {son, father pair} also implies that $f(father) > f(any\ other\ neighbor\ of\ the\ son\text{'}s)$. In both cases a cycle means there is a contradiction via transitivity of "$>$".

Since only one OAT can be generated from an ordering, the expected mean pathlength of an OAT from, say, the boundary distribution is well defined. This is not true for BATs because usually many BATs can be generated from a single ordering. To resolve the ambiguity, we adopt the convention that a BAT is to be randomly generated from an ordering by choosing from among the possible fathers with equal probability, for each vertex. We can now discuss the expected mean pathlengths of OATs and BATs from the coboundary and other distributions.

CHAPTER 2.
SOME PROPERTIES OF OATS AND BATS

2.1 Best Case Trees

The purpose of this chapter is to explore some combinatorial properties of OATs and BATs. The later chapters do not depend logically on any of this material, but we hope that it will serve to give the reader more familiarity with the structures defined in Chapter 1. The first topic we deal with is the best case for a local improvement algorithm.

In an instance of an LG problem with best possible structure, the algorithm would decrease the Hamming distance to the optimum point at every iteration. If the origin were the optimum, we would always change ones to zeroes but never zeroes to ones. Since on the average the starting vertex will have half of its coordinates equal to one, we would expect the average number of iterations to be about $n/2$. We have defined the pathlength of the root to be 1 (it takes one iteration to verify optimality), so the exact value of the lowest possible mean pathlength is in fact $1 + n/2$.

One tree structure which reflects this best case situation is the binomial tree (see Knuth, 1973). The binomial tree of order n is constructed inductively by appending the binomial trees of order $n-1, n-2, \ldots, 0$ to a root vertex. Figure 2.1 shows the binomial trees of orders one, two, and three.

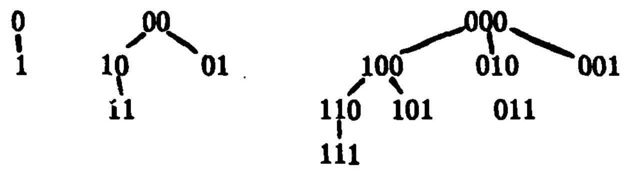


Figure 2.1

One of the orderings that generates a binomial tree can be found by listing the vertices in the tree level by level (starting at the root), and going from left to right on each level. There are of course other orderings that would yield the same binomial tree. In general one can employ the topological sort method (as described in Knuth, 1973) to produce other orderings which yield the same OAT as a given ordering.

The binomial tree is not the only possible OAT with smallest mean pathlength. In Figure 2.2 we show an OAT with mean pathlength $1 + n/2$ for $n = 3$ which is not a binomial tree.

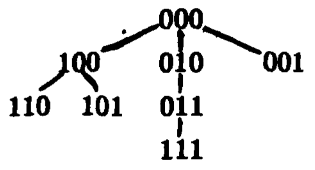


Figure 2.2

The best possible mean pathlength of a BAT is of course also $1 + n/2$. The example of a BAT that fails to be an OAT, Figure 1.4, is one of many such trees. The total number of best case BATs is fairly large. Since there are $\binom{n}{k}$ nodes positioned k levels below the root, and each of these nodes has k possible fathers on the next level up, there exist

$$\prod_{k=1}^{n} k^{\binom{n}{k}}$$

best case BATs whose root is 00...0, and there are 2^n times as many altogether.

2.2 Worst Case Trees

In contrast with the best case, the problem of finding the largest possible height or mean pathlength of an OAT is an unsolved combinatorial problem. Even for small values of n it is not a trivial problem to find an OAT of greatest height; to give an idea of how quickly OATs become complicated we show a worst case OAT of order 4 in Figure 2.3.

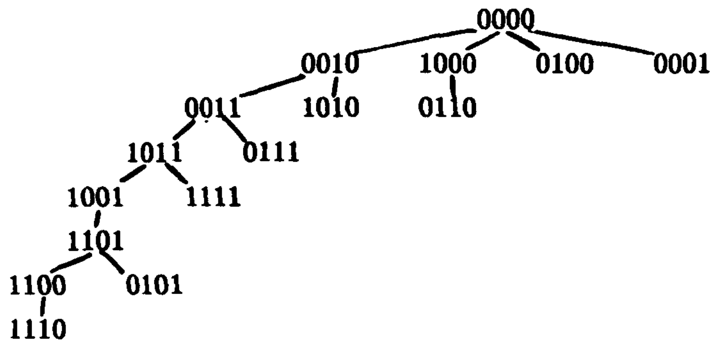


Figure 2.3

Any path from a vertex to the root of an OAT must satisfy what we call the *grandfather clause*: the vertex may not be adjacent to any vertices in the path except its father. If the vertex were adjacent to its great grandfather, for instance, it would not be the son of its optimal adjacent point, since the great grandfather must have a better function value than the father. Therefore, a vertex in a path of an OAT is adjacent to its father and its son, but not to any other vertices in the path. This property is similar to the idea of a snake-in-a-box. A *n-dimensional snake-in-a-box* is defined to be a tour, or circuit, of vertices on an n-cube with the property that every vertex in the tour is adjacent to exactly two other vertices in the tour (the ones immediately preceding and succeeding it). If we remove one vertex from a snake-in-a-box, we have a path that satisfies the grandfather clause. It is easy to construct an OAT which contains some particular path: for instance, make the vertices in the path the first vertices in the ordering. The other vertices can be placed below them in the ordering in any way that does not violate the LG property, and the OAT produced from such an ordering will contain the given path.

Therefore, given a snake-in-a-box containing L vertices, we can always construct an OAT whose height is at least $L - 1$. Victor Klee (Klee, 1970) has shown that, for $n \geq 6$, there exists a snake-in-a-box of length

$$\frac{7 \times 2^n}{4(n-1)}.$$

(The greatest length is unknown.) Subtracting one from this number gives the lower bound on the maximum height of an OAT which was stated in Chapter 1.

BATs are less tightly constrained than OATs and hence are often easier to analyze. In particular, BATs need not satisfy the grandfather clause. The worst case BAT is a single path; its height is 2^n, and it is exemplified by the Gray code (see Reingold *et al*, 1977, and Cottle, 1978).

2.3 The Local-Global Property

The total number of orderings of order n is $(2^n)!$. For very small values of n the orderings can be enumerated to discover the number of them that are LG. When n equals two, 2/3 of them are LG; when n equals three, 3/14 of them are LG. As n increases, this proportion decreases rapidly. However, the sequence does not appear to follow any simple combinatorial pattern. We can estimate the proportion of orderings that are LG in the following way: an ordering is LG if every vertex in the ordering, except the root, has a neighbor above it. Thus the probability of a random ordering's being LG is the probability of the intersection of these events,

$$Prob\left[\bigcap_{k=2}^{2^n} (\textit{the kth vertex is not a local optimum})\right]. \tag{2.3.1}$$

We make the (false) simplifying assumption that these events are independent and approximate (2.3.1) by

$$\prod_{k=2}^{2^n} P(k), \tag{2.3.2}$$

where $P(k)$ denotes the probability that the kth vertex in the ordering is not a local optimum. Next we approximate the $P(k)$ by

$$P(2^n - k) = 1 - \binom{k}{n} \Big/ \binom{2^n - 1}{n} \approx 1 - (k/2^n)^n. \tag{2.3.3}$$

The left hand side of (2.3.3) is one minus the probability that the n neighbors will be arranged in the k lowest positions in the ordering, which is one minus the probability that the vertex in the $k+1$st lowest position is a local optimum. Substituting (2.3.3) into (2.3.2), the estimate of the probability of occurence of the LG property becomes

$$\begin{aligned}\prod_{i=0}^{2^n-2}\left(1-(i/2^n)^n\right) &= \exp\left(\sum_{i=0}^{2^n-2}\log\left(1-(i/2^n)^n\right)\right)\\ &\approx \exp\left(-\sum_{i=1}^{2^n-1}(i/2^n)^n\right) \approx \exp\left(\frac{-(2^n)^{n+1}/(n+1)}{(2^n)^n}\right)\\ &= e^{-2^n/n+1} \qquad (2.3.4)\end{aligned}$$

This estimate indicates that the LG property is quite special. The chances of a random ordering's being LG become vanishingly small as n gets large. Experimental results suggest that the estimate is somewhat high. This makes sense because the estimate is based on the assumption that the events, "the kth vertex is not a local optimum", are independent. In fact these events are negatively correlated, for if a vertex is not a local optimum, its neighbors have a slightly higher chance of being one. The effect of the simplifying assumption, therefore, would be to increase the estimate.

We should point out that the assumption that all orderings are equally likely to occur is not necessarily very realistic. In many problems, function values of adjacent vertices tend to be relatively close to one another. A more realistic simplifying assumption, then, might be that the function values of a point's neighbors are independently distributed, each with an equal chance of being better or worse. In this case, the probability that a vertex is a local optimum is 2^{-n}, so the probability that the ordering is LG is approximately $1/e$. Unfortunately this assumption is not self-consistent, but it is interesting because it suggests a relationship between problems that are LG and problems in which adjacent vertices tend to have similar function values.

We have seen that the concept of the boundary gives us a simple formulation of the LG property: an ordering is LG if and only if its ith vertex is in the boundary of the first $i-1$ vertices, for $i = 2, \ldots, 2^n$. Fix n. If $B(i)$ is a number equal to the size of the smallest possible boundary of a set of i points, then

$$\prod_{i=1}^{2^n-1} B(i) \tag{2.3.5}$$

must be a lower bound on the total number of LG orderings. In the next chapter we derive the following bounds:

$$\begin{aligned} B(i) &\geq \binom{n}{k+1}, \qquad P_k \leq i < P_{k+1}, \quad k \leq \lfloor \frac{n-1}{2} \rfloor \\ &\geq \binom{n}{k+2}, \qquad P_k \leq i < P_{k+1}, \quad k \geq \lfloor \frac{n-1}{2} \rfloor \end{aligned}$$

where P_k denotes the sum of the first $k+1$ binomial coefficients. Substituting into (2.3.5), we get

$$\begin{aligned} \textit{number of LG orderings} &> \prod_{k=1}^{\lfloor \frac{n+1}{2} \rfloor} \binom{n}{k}^{\binom{n}{k}} \times \prod_{k=\lfloor \frac{n+1}{2} \rfloor}^{n} \binom{n}{k+1}^{\binom{n}{k}} \\ &> 2^{\sum_{k=1}^{\lfloor \frac{n-1}{2} \rfloor} (2k-1)\binom{n}{k}} \\ &> 2^{2^n}. \end{aligned} \tag{2.3.6}$$

Although (2.3.6) is a conservative lower bound, it is clear that even for $n = 6$ it would not be feasible to perform an exhaustive enumeration of LG orderings.

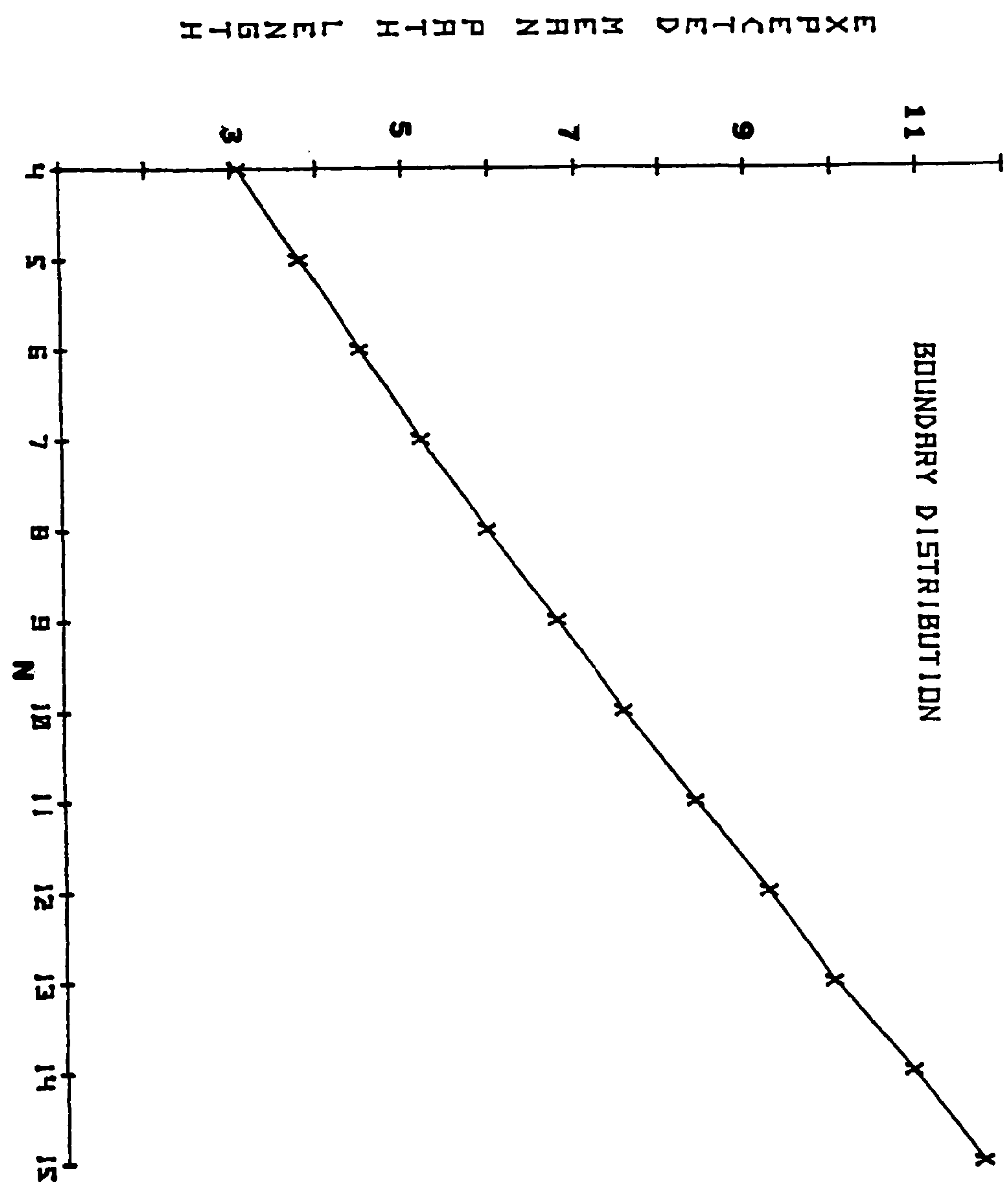
CHAPTER 3.
BOUNDS ON EXPECTED MEAN PATHLENGTH

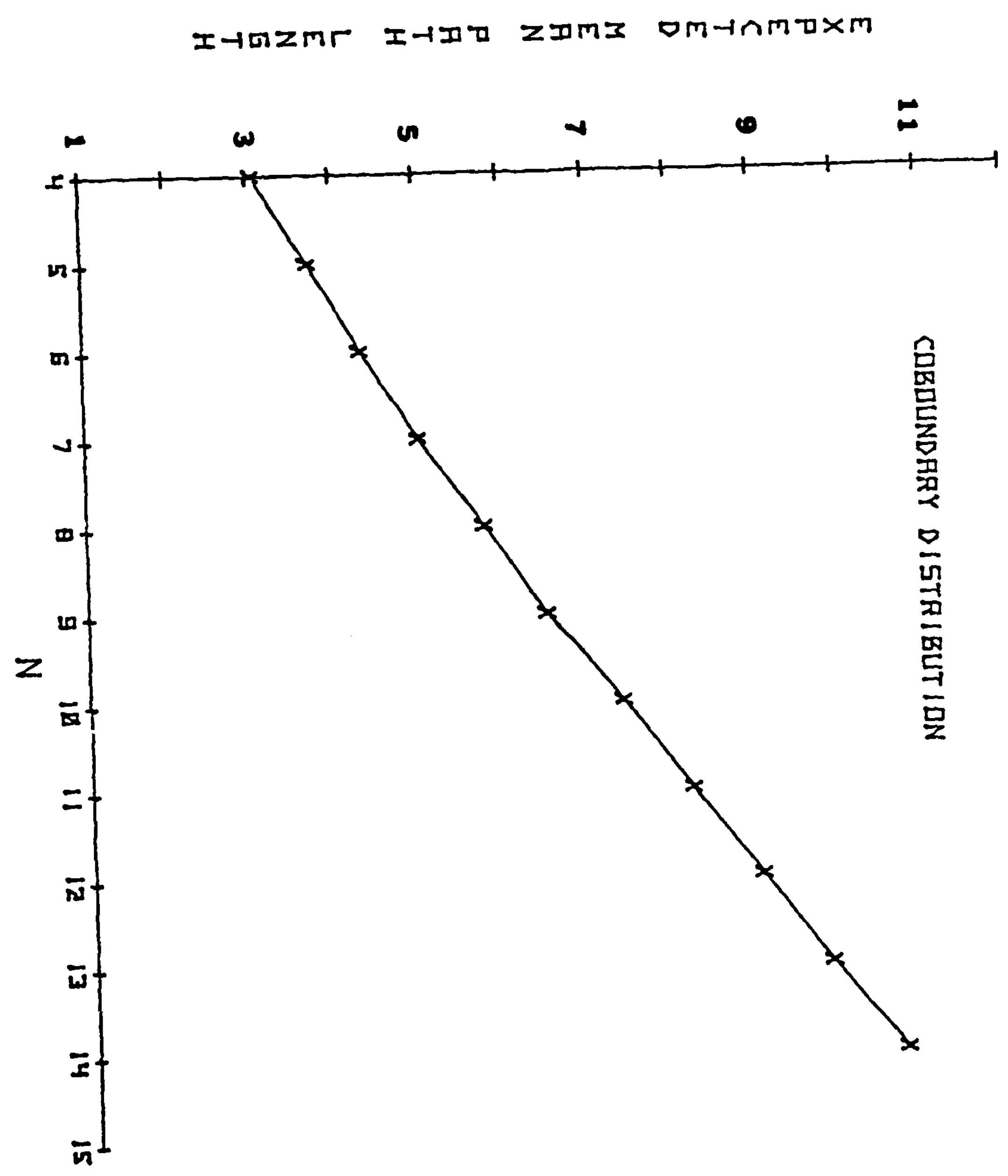
3.1 Empirical results

3.1.1 Linearity of expected mean pathlength

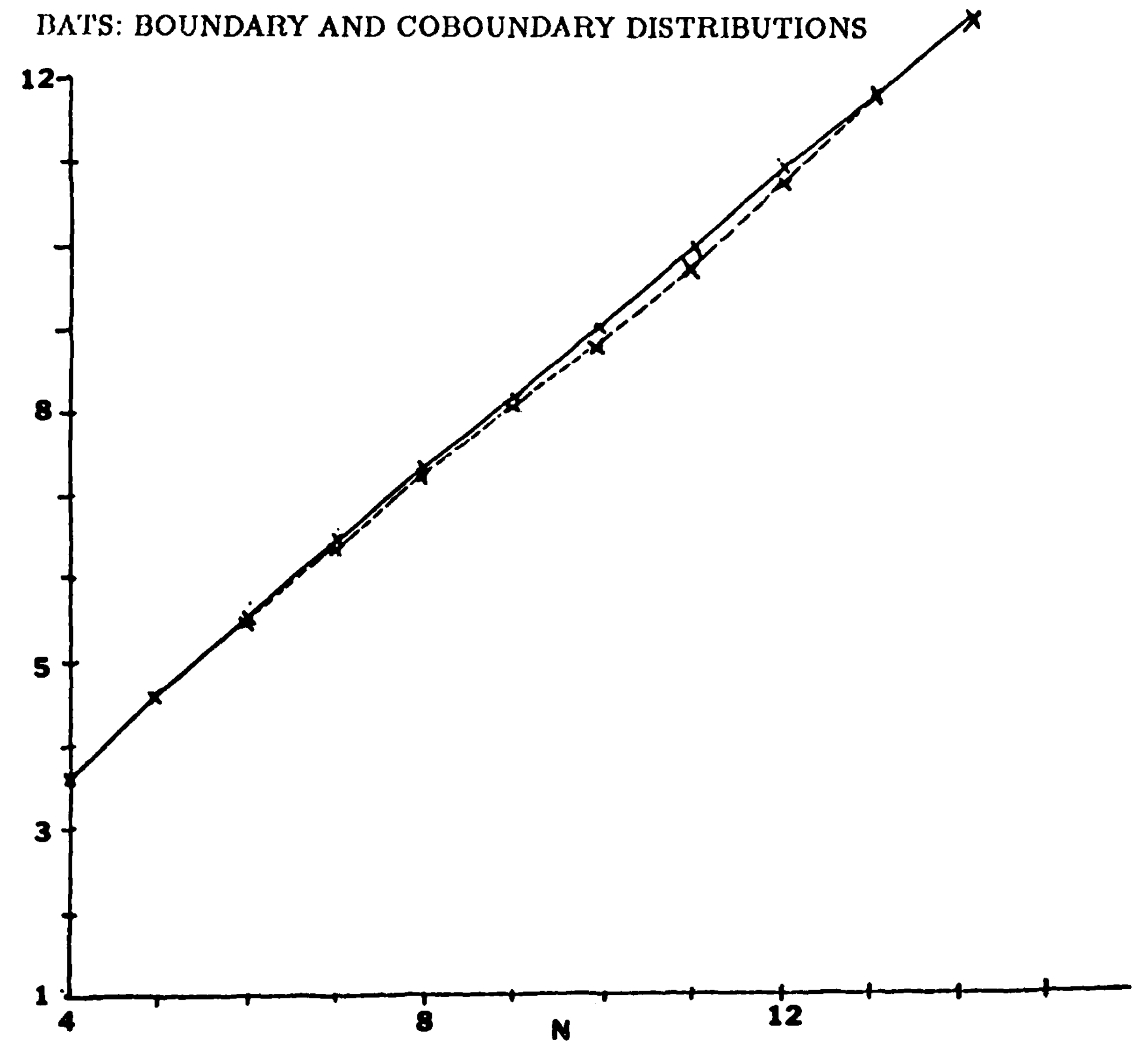
First we briefly present some empirical results. From imulations performed by Craig Tovey, Frank Heartney, and Charles Fay, it appears that the expected mean pathlength of both OATs and BATs is linear in n for $n = 4, \ldots, 15$ under all three notions of randomness: boundary, coboundary, and all LG orderings equally likely (see the graphs on the following pages). Bad cases – large heights – appear to be extremely rare. This is the fundamental empirical result. Though over 200,000 OATs altogether were generated, the largest mean pathlengths and heights seen were just two to three times the average. The expected value of the mean pathlengths appears to be linear in n although there exist cases that are $O(2^n/n)$. The "bad" cases evidently are so rare that it doesn't matter much which underlying distribution is used: the expected mean pathlength remains small. The empirical results lead us to conjecture that expected mean pathlength is less than n.

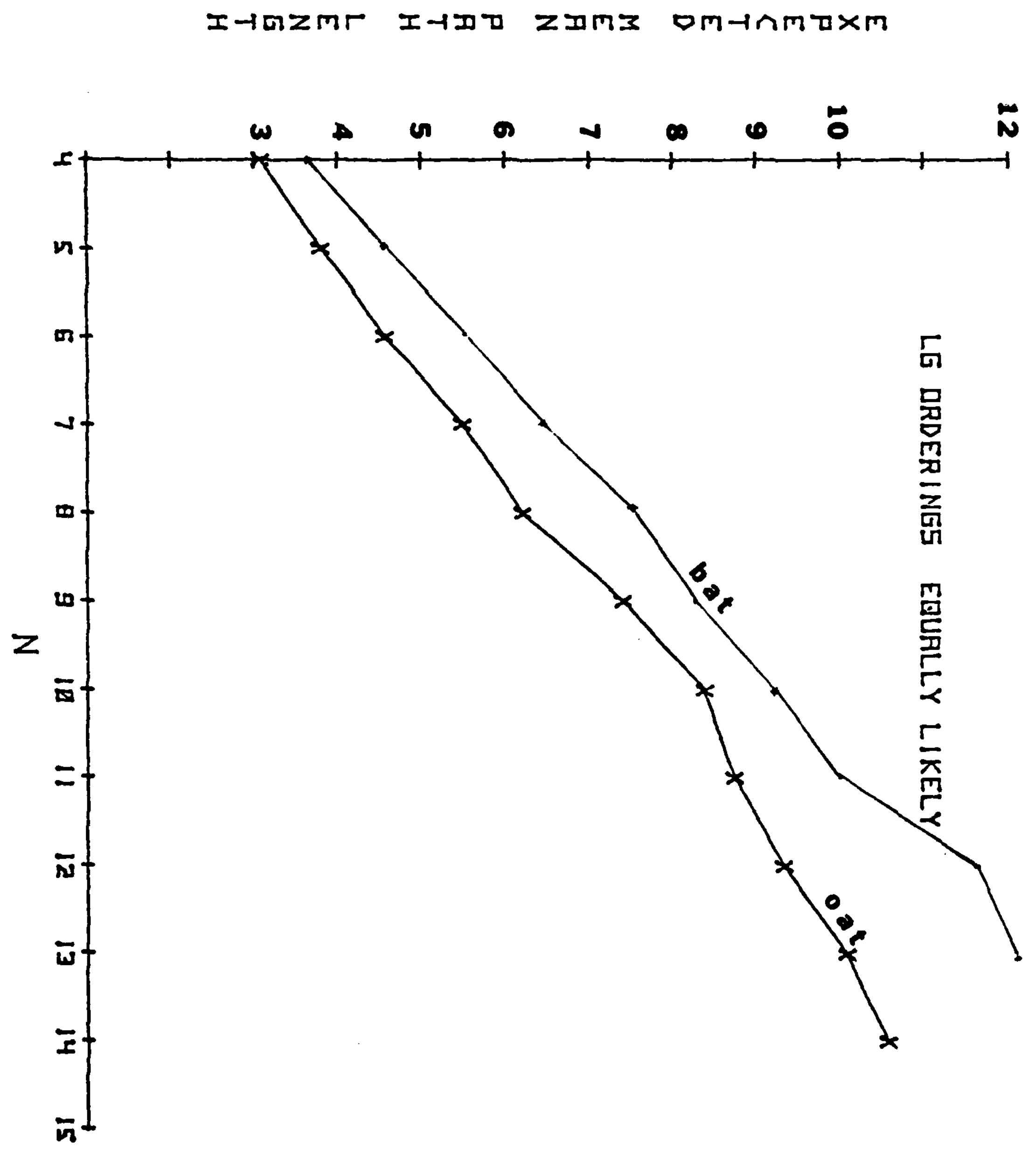
Conjecture 3.1 *Under any of the distributions, boundary, coboundary, and LG orderings equally likely, the expected mean pathlength of both OATs and BATs is less than n. Equivalently, the expected number of iterations of an optimal adjacency or better adjacency algorithm is less than n with any of these distributions.*

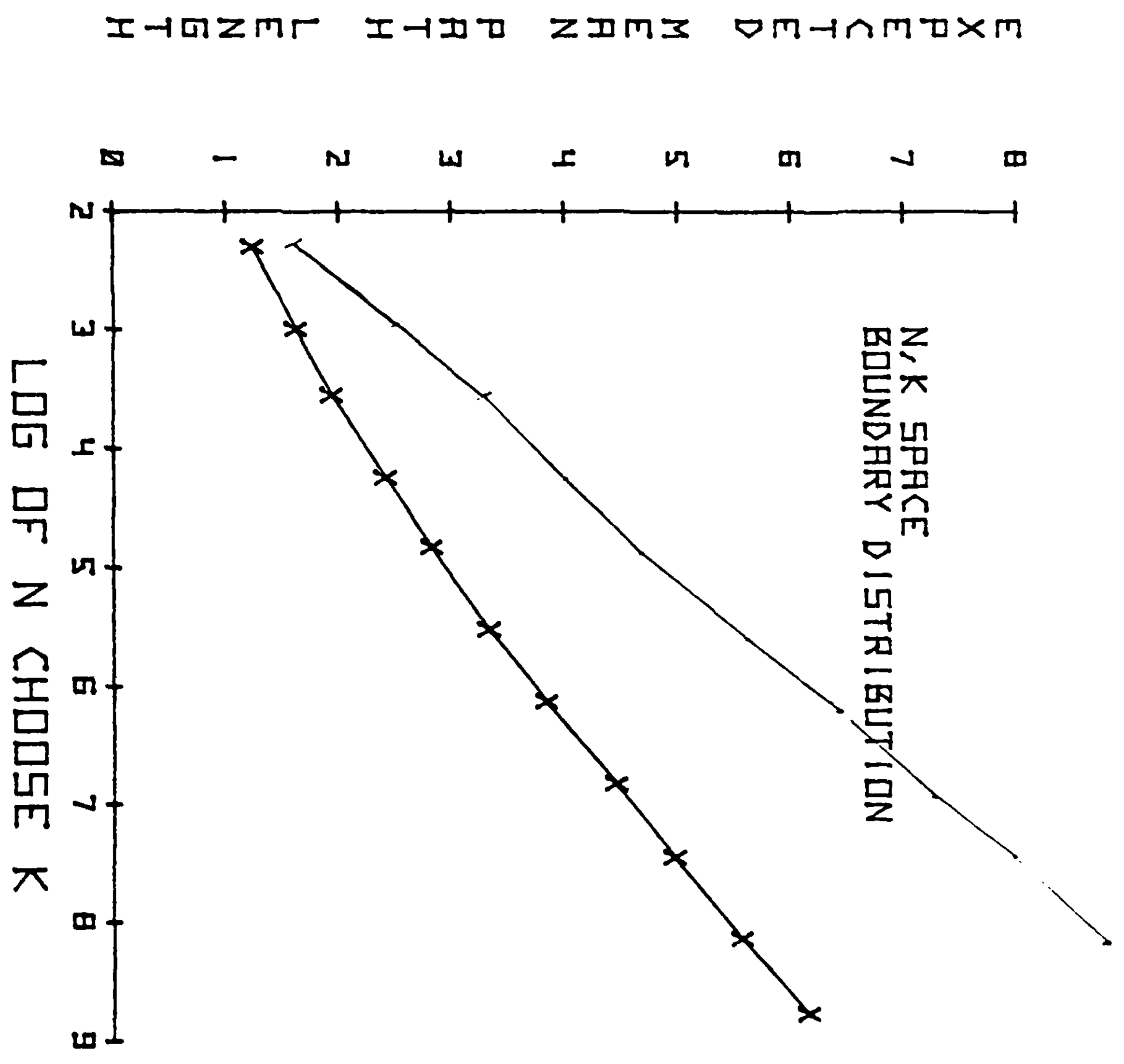

BOUNDARY DISTRIBUTION
EXPECTED MEAN PATH LENGTH
3
5
7
9
11
N
4
5
6
7
8
9
10
11
12
13
14
15


COBOUNDARY DISTRIBUTION
EXPECTED MEAN PATH LENGTH
N
1
3
5
7
9
11
4
5
6
7
8
9
10
11
12
13
14
15

BATS: BOUNDARY AND COBOUNDARY DISTRIBUTIONS




LG ORDERINGS EQUALLY LIKELY
EXPECTED MEAN PATH LENGTH
N
bat
oat
3
4
5
6
7
8
9
10
12
4
5
6
7
8
9
10
11
12
13
14
15


EXPECTED MEAN PATH LENGTH
0
1
2
3
4
5
6
7
8
N,K SPACE
BOUNDARY DISTRIBUTION
2
3
4
5
6
7
8
9
LOG OF N CHOOSE K

3.1.2 OATs versus BATs

The empirical results cited above allow us to compare the relative merits of "optimal adjacency" and "better adjacency". Not surprisingly, OATs have a smaller average mean pathlength: approximately $.79n - .2$ for OATs versus $.92n - .1$ for BATs. We can expect, therefore, that choosing the best neighbor rather than any better neighbor will lead to about a fifteen percent reduction in the number of iterations, on the average. This estimate is pertinent to the design of local improvement algorithms for integer programming (Hillier, 1969). As we will discuss in Chapter 5, to apply the model to the Simplex Method for linear programming, the model can be modified so that it includes only a subset of the hypercube vertices. When this is done, the difference between OATs and BATs increases to about fifty percent, which is consistent with experimental results of 38 to 67 percent found in (Cutler and Wolfe, 1963). Though more iterations may be required, the cost of a single iteration is apt to be considerably less in the case of better adjacency because fewer function evaluations are required. We face a tradeoff between the total number of iterations and the cost of a single iteration. It seems likely that better adjacency (or some improved variant of it, such as the best gradient approach used in the Simplex Method) will usually yield the more efficient algorithm. Which alternative we choose will depend on the nature of the specific problem.

3.2 Theoretical Results

Theorem 3.2 *For the boundary and coboundary distributions, the expected height of a BAT is less than $en^2 \log n$. The same bound holds for OATs with the boundary distribution.*

Corollary. *The expected number of iterations of the Better Adjacency algorithm, with respect to either the boundary or coboundary distribution, is less than $en^2 \log n$. The same bound holds for the Optimal Adjacency algorithm with respect to the boundary distribution.*

The rest of this section is devoted to the proof of this theorem. Since the proof at times seems to have little to do with the theorem, it may be helpful to give a general idea of it first.

Imagine being at some stage of a process which randomly generates an LG ordering and an associated tree (OAT or BAT). At each step a vertex is selected from the boundary, assigned a father, and attached to the tree; the vertex has a pathlength one greater than its father's. The height of the ith vertex in the ordering is denoted by H_i. Then the boundary list is updated and the process iterates. We want to know how high the tree is going to get. This depends on which vertices tend to be fathers. For all we know, though, it could be that the vertices with greatest height tend to be fathers. If so, then we will likely get a long, skinny tree. But suppose we knew that, at each iteration, some number, say 100, of all the nodes in the tree had an equal chance of being the new father. What would this piece of information tell us? In the worst case the 100 nodes would be the 100 lowest nodes in the tree generated so far. Thus it would imply that the expected height of our tree could be no greater than the expected height of a tree produced by repeatedly attaching a son to a node randomly chosen from among the hundred lowest nodes in the tree. This new process is less complicated than the original one, and it turns out to be possible, though not trivial, to compute its rate of growth, and thus a bound on OATs and BATs.

The proof of the theorem, therefore, consists of three parts. The first part derives a lower bound on the number of vertices in the boundary. This will yield a lower bound on the number of potential fathers, because no potential father can have more than $n-1$ neighbors in the boundary. The second part proves the intuitively obvious fact that the "new" process does indeed produce trees with greater mean pathlength than the "old" BAT and OAT producing processes. The third part of the proof is the computation of the rate of growth of the "new" process.

Recall that a hypergraph H is a set of points, called vertices, together with a collection of subsets of the vertices, and that these subsets are called edges. In the case of a graph, all the edges contain exactly two vertices; in the more general hypergraph the edges can vary in count, some possibly consisting of one vertex, others with several. For example, a hypergraph on the 3 vertices a, b, and c might contain the edges $\{a\}$ and $\{a, b, c\}$. As another example, the hypergraph on n vertices which contains all possible edges will have 2^n of them, and will be a representation of the set of all subsets of the n vertices.

If we think of the vertices of the hypergraph as corresponding to each of the components (or dimensions) of the n-cube, an edge of the hypergraph as being a vertex of the n-cube, and adjacent pairs of hypergraph edges as corresponding to edges of the n-cube, then we see that the Kruskal-Katona theorem that follows is a statement about sets of vertices of the n-cube with minimal size boundaries. We define a *minimal boundary set* of cardinality i to be a collection of i vertices of the n-cube whose boundary is as small as possible for a set of i vertices.

If H is a hypergraph on n vertices having M edges, then the boundary of H is the set of all edges not in H that differ from members in H in only one vertex. Order the vertices of H from 1 to n and denote an edge e by a sequence of zeroes and ones, the i-th entry being one when e contains the i-th vertex. Define $w(e)$ to be the $n+1$-digit number which is this sequence with a leftmost digit appended so that the sum of all the digits equals n.

Theorem 3.3 *(Kruskal, Katona) A hypergraph on n vertices having M edges that minimize its boundary can be formed by choosing the M edges having largest w values.*

Proof. See (Kleitman, 1979), pp. 47–48.

Boundary Theorem 3.4 *Let P_k denote*

$$\sum_{j=0}^{k} \binom{n}{j},$$

the sum of the first $k+1$ binomial coefficients and let

$$i = P_k.$$

Then the size of the smallest boundary of a set of size i is $\binom{n}{k+1}$; If i instead satisfies

$$P_k \ < \ i \ < \ P_{k+1}$$

then a lower bound on the boundary size of a set of size i is given by

$$\begin{array}{ll} \binom{n}{k+1}, & if\, k \leq \frac{n-3}{2}; \\ \binom{n}{k+2}, & if\, k \geq \frac{n-3}{2}. \end{array}$$

When $k = \frac{n-3}{2}$, the bounds for the two cases are equal.

Proof. According to the construction of the minimal boundary set given by the Kruskal-Katona theorem, all vertices (of the n-cube) with j ones must be picked before any vertex with $j+1$ ones, because their w values have a larger leading digit. That is, if the minimal boundary set given by the theorem contains a vertex with $j+1$ ones, it must contain all vertices with j ones. Since the number of vertices with j ones equals $\binom{n}{j}$, the minimal boundary set of cardinality i is the set of all vertices with k or fewer ones when i equals P_k, the sum of the first $k+1$ binomial coefficents. The boundary of this set consists of all vertices with exactly $k+1$ ones. There are $\binom{n}{k+1}$ such points, so we have proved the first part of the theorem.

We need the LYM (Lubell, Yamamoto, and Meshalkin) inequality to prove the rest of the theorem. The possible edges of a hypergraph form a partially ordered set by inclusion. Recall that an antichain F is a collection of these edges such that no member of F contains another. If we denote the number of members of F of cardinality j by f_j, then the LYM inequality (Kleitman, 1974) states that

$$\sum_{j=0}^{n} \frac{f_j}{\binom{n}{j}} \leq 1.$$

Now suppose that i is some intermediate value between P_k and P_{k+1}. The Kruskal-Katona set contains all vertices with k ones and some vertices with $k+1$ ones. The set's boundary obviously includes all vertices with $k+1$ ones — called "*k+1*-vertices" for short— that are not in the set itself, as well as those $k+2$-vertices that are neighbors of the $k+1$-vertices in the set. Let s be the number of $k+1$-vertices in the set, so $s = i - P_k$. Let z equal the number of $k+2$-vertices not in the set's boundary. The $k+1$-vertices in the set, together with the $k+2$-vertices not in the boundary, form an antichain. Then by the LYM property,

$$\frac{s}{\binom{n}{k+1}} + \frac{z}{\binom{n}{k+2}} \leq 1.$$

If k is less than or equal to $(n-3)/2$ this implies that

$$\binom{n}{k+2} - z \geq s\frac{\binom{n}{k+2}}{\binom{n}{k+1}} \geq s$$

The left hand side of the above is the number of $k+2$-vertices in the boundary, so the total number of elements in the boundary is at least $\binom{n}{k+1}$. Similarly, if $k \geq (n-3)/2$, the LYM inequality yields

$$\binom{n}{k+1} - s \geq z\frac{\binom{n}{k+1}}{\binom{n}{k+2}} \geq z$$

The left hand side of this relation is the number of $(k+1)$-vertices in the boundary, and therefore the size of the boundary is at least $\binom{n}{k+2}$. Note that when k equals $\frac{n-3}{2}$, the number n must be odd, so

$$\binom{n}{k+1} = \binom{n}{\frac{n-1}{2}} = \binom{n}{\frac{n+1}{2}} = \binom{n}{k+2}$$

and the bounds for the two cases coincide. ∎

Consider the process which randomly generates an OAT according to the boundary distribution, and let the random variables $H_1, \ldots, H_{2^n}$ denote the heights of the nodes in the ordering. By convention the "height" of the top node or root is

1. Then H_1 and H_2 always take the values 1 and 2 respectively, H_3 takes the values 2 and 3 with equal probability, and the distributions of the later H_i are increasingly complicated.

If X and Y are two random variables with cumulative distribution functions $F_x(t)$ and $F_y(t)$ respectively, then X is said to stochastically dominate Y, written $X \succeq Y$, if $F_x(t) \leq F_y(t) \forall t$. It is obvious that stochastic dominance is transitive and that if $X \succeq Y$ then $E[X] \geq E[Y]$ where $E[\,]$ denotes expectation. To extend this concept to sequences such as the H_i we say that the sequence of random variables $X = X_1, X_2, \ldots$ stochastically dominates $Y = Y_1, Y_2, \ldots$ iff $X_i \succeq Y_i \; \forall i$, and we write $X \succeq Y$.

We will now describe a stochastic process whose outcomes will stochastically dominate the H_i. The process is called the *largest k process* and is denoted by L^k. Let $k = k_1, k_2, \ldots$ be a sequence of positive integers. The sequence of random variables $L^k = L_1^k, L_2^k, \ldots$ is iteratively produced in the following way: $L_1^k = 1$; given the values of $L_1^k, \ldots, L_{i-1}^k$, one of the k_i largest of them is selected at random (each with equal probability) and its value plus one is given to L_i^k. For example, if k is a sequence of ones, that is if $k_i = 1 \; \forall i$, then L_i^k will equal i for all i. As another example, suppose that $k_1 = k_2 = 1$, and $k_3 = k_4 = 2$. Then L_1^k will equal 1, L_2^k will equal 2, L_3^k will take the values 2 and 3 with equal probability, and L_4^k will take the value 3 with probability 3/4 and the value 4 with probability 1/4. (It is sometimes convenient to let k_i be greater than $i - 1$; in these cases we are interested only in the long term behavior of the L_i and we will be able to assume the existence of $L_0, L_{-1}, L_{-2}, \ldots$ each with value zero.)

We now return to the random OATs produced by the boundary distribution and the random heights of its orderings $H_1, \ldots, H_{2^n}$. Let $\mathcal{V} = \mathcal{V}_1, \ldots, \mathcal{V}_{2^n}$ be random variables which are the vertices in the ordering generated, so $\mathcal{V}_i$ is the ith vertex in the ordering. Consider any partial realization of the OAT generating process: suppose the first i vertices in the ordering have been chosen and denote the list of vertices by $v = v_1, \ldots, v_i$.

Suppose further that the random variables $H_1, \ldots, H_i$ have values $h_1, \ldots, h_i$ respectively. For simplicity of notation, we use $\mathcal{V} = v$ to denote the condition

$$\mathcal{V}_j = v_j; \qquad j = 1, \ldots, i$$

and similarly, we let $H = h$ denote the condition

$$H_j = h_j; \qquad j = 1, \ldots, i.$$

What can we say about the conditional distribution of H_{i+1}, given $\mathcal{V} = v$, $H = h$? When i is greater than one, any vertex v_j, $1 \leq j \leq i$, can have at most $n - 1$ neighbors in the boundary of v. Let $\mathcal{B}(v)$ denote the boundary of v. According to the boundary distribution, every member of $\mathcal{B}(v)$ has an equal probability of becoming the value of $\mathcal{V}_{i+1}$. Therefore, for any particular j, the probability of v_j being the father of v_{i+1} is less than or equal to

$$\frac{n-1}{|\mathcal{B}(v)|} \leq \frac{n-1}{B(i)},$$

where $B(i)$ is the lower bound on the boundary size given by the boundary theorem.

Assume inductively that we have constructed a sequence $\mathcal{L} = \mathcal{L}_1, \ldots, \mathcal{L}_i$, such that $\mathcal{L}_j \geq H_j$, for all $j = 1, \ldots, i$, and such that $\mathcal{L}$ is distributed as a largest k process with $k_j = B(j)$. Given $H = h$, $V = v$, it is easy to construct $\mathcal{L}_{i+1} \geq H_{i+1}$ so that $\mathcal{L}_{i+1}$ takes the largest $\lfloor B(i)/(n-1) \rfloor$ values in the $\mathcal{L}$ sequence with equal probability. This completely defines the distribution of $\mathcal{L}_{i+1}$ since we have defined its conditional distribution for all conditions $H = h$, $V = v$. Of course, the distributions of H and $\mathcal{L}$ are not independent. We may not know the probability of a particular event $H = h$, $V = v$. We do know, however, that $\mathcal{L}$ is distributed as a largest k process with $k_i = \lfloor B(i)/(n-1) \rfloor$. This is true because all of the conditional distributions have that characteristic. We have therefore proved the dominance lemma below for OATs.

Dominance Lemma 3.5 *Suppose $B(i)$ is a lower bound on the size of the boundary of i vertices. Let $k = k_1, \ldots, k_{2^n}$ be the vector of integers such that*

$k_i = \max[1, \lfloor B(i)/(n-1) \rfloor]$ *and let* H *denote the sequence of random variables which are the heights of the nodes in an OAT from the boundary distribution. Then* $L^k \succeq H$*. Moreover, the same result holds for a BAT from either the boundary or coboundary distribution.*

Proof. The case of BATs from the boundary distribution is almost exactly the same the above. The key fact, again, is that the probability of v_j being the father is not more than

$$\frac{n-1}{B(i)}$$

for any particular j. The only difference is that in this case, it is necessary to specify the events by $V = v$, $H = h$. In the case of OATs, it would have sufficed to specify only $V = v$, since with OATs the choice of vertices in the ordering completely determines the pathlengths.

The case of coboundary BATs is similar. If S is a subset of vertices of the n-cube, we define the *coboundary* of S as the set of all pairs (x, y) such that x is in S, y is not in S, and x and y are adjacent. In this case we let S equal v, so the coboundary consists of all pairs (x, y) such that x is in the partially constructed ordering, y is in the boundary, and x and y are adjacent. To generate the BAT we choose among the coboundary pairs with equal probability. Again, no father can appear in more than $n-1$ pairs, and since there are at least as many pairs as boundary elements, the result must be stochastically dominated by L^k. This completes the proof of the lemma. ∎

Now we find the expected behavior of L^k. The value of $B(i)$ and hence k_i remains the same as i ranges between two consecutive partial sums of the binomial coefficients. While the value of k_i is constant and equal to say, m, the largest k process can be thought of as a climbing process, where there are m balls distributed contiguously on at most m levels. There are infinitely many empty levels above for the balls to climb up to. At each iteration, a ball is selected at random: a new ball is added to the level above the selected ball, and one ball is removed from the lowest non-empty level. ;imulation results by the author suggested that this process climbs at a rate approaching e/m (where e is the natural logarithmic constant) for large m. Incomplete proofs were given by (Keller, 1980; Wiener, 1980). This conjecture was finally proved by David Aldous and James Pittman:

Theorem 3.6 *Let μ_m denote the expected rate of growth of the climbing process when there are m balls. Then $m\mu_m$ is less than e and asymptoticaly approaches e as $m \to \infty$. Equivalently, let m be a positive integer and let M be a sequence of m's. Then the expected rate of growth, μ_m, of the sequence L^M, is less than e/m and $m\mu_m \to e$ as $m \to \infty$.*

For a complete proof of this result, see (Aldous and Pittman, 1980). Here we present that part of Aldous and Pittman's work which shows that the expected rate of growth, μ_m, is less than or equal to e/m.

At any time t there exists a finite colony of particles among sites 0, 1, 2, ..., the sites forming some interval $[0, l]$. The colony evolves by individual particles independently giving birth at rate 1. The "daughter" particle gets placed at the site to the right of the "mother" particle. Let $X_i(t)$ be the number of particles at site i at time t. Let

$$X(t) = \{X_0(t), X_1(t), \ldots\}.$$

We can think of $X(t)$ as a countably valued Markov chain. Note that the $X(t)$ do not give a complete description of the process since they ignore geneology.

The total number of particles at time t,

$$N(t) = \sum_{i \geq 0} X_i(t)$$

evolves as a simple birth process. Therefore,

$$\frac{dN(t)}{dt} = N(t)$$

and

$$E(N(t)) = e^t.$$

The front of the colony at time t is the largest index i such that $X_i(t) \geq 1$, and is denoted by $f(X(t))$. We will prove that

$$\limsup_{t\to\infty} \frac{f(X(t))}{t} \leq e \quad a.s. \tag{3.2.1}$$

Let

$$m_i(t) = E(X_i(t))$$

equal the expected number of particles at site i at time t. Then $m_0(t) = 1$ if we start with one particle at zero. Also,

$$\frac{d\,m_i(t)}{dt} = m_{i-1}(t); \qquad i \geq 1.$$

Solving these differential equations gives

$$m_i(t) = \frac{t^i}{i!}. \tag{3.2.2}$$

Now let γ be some constant greater than e. We have

$$P(f(X(t)) > \gamma t) = E(1_{\{f(X(t))>\gamma t\}}) \leq E\left(\sum_{i>\gamma t} X_i(t)\right). \tag{3.2.3}$$

By definition and (3.2.2),

$$E\left(\sum_{i>\gamma t} X_i(t)\right) = \sum_{i>\gamma t} m_i(t) = \sum_{i>\gamma t} t^i/i!.$$

Combining (3.2.3) with the above yields

$$\sum_{t=1}^{\infty} P(f(X(t)) > \gamma t) \leq \sum_{t=1}^{\infty} \sum_{i>\gamma t} t^i/i! \leq \sum_{i=1}^{\infty} \frac{1}{i!} \sum_{t<i/\gamma} t^i$$

$$\leq \sum_{i=1}^{\infty} \frac{1}{i!}(i/\gamma + 1)^{i+1} \tag{3.2.4}$$

By Stirling's formula, the summation term in (3.2.4) can be approximated by

$$e^{\gamma}(i/2\pi)^{1/2}(e/\gamma)^{i}.$$

Therefore, the sum in (3.2.4) converges. By the Borel-Cantelli Lemma,

$$f(X(t)) \leq \gamma t$$

for all sufficiently large integers t, so (3.2.1) is true.∎

We can now prove theorem 3.2, which we stated at the beginning of the section. The expected height reached by the L_i^k for $i = 2$ up to the sum of the first $\lfloor\frac{n-1}{2}\rfloor$ binomial coefficients is

$$\sum_i e(n-1)/B(i) = e(n-1)\sum_{j=1}^{\lfloor\frac{n-1}{2}\rfloor}\binom{n}{j}\bigg/\binom{n}{j} = e(n-1)\lfloor\frac{n-1}{2}\rfloor.$$

The expected increase in L_i^k as i then increases to $2^n - 1$ is

$$e(n-1)\sum_{j=\lfloor\frac{n-1}{2}\rfloor}^{n-1}\binom{n}{j}\bigg/\binom{n}{j+1} \approx e(n-1)(n\log n - n/2).$$

The expected increase in L_i^k as i increases from 1 to 2 and from $2^n - 1$ to 2^n is no more than 2. The sum of these three increases is less than $en^2 \log n$, which is therefore an upper bound on the expected height of boundary OATs and BATs, and coboundary BATs. ∎

CHAPTER 4.
ADDITIONAL BOUNDS ON AVERAGE MEAN PATHLENGTH

4.1 Coboundary OATs

The proof of the dominance lemma fails in the case of coboundary OATs. Both common sense and simulation trials suggest that coboundary OATs must have lower bounds than coboundary BATs, but we have been unable to prove this is indeed the case. It is not hard to see, however, that if m equals $B(i)/n(n-1)$ instead of $B(i)/(n-1)$, the L^K process must bound the H_i values of coboundary OATs. This puts an additional factor of $1/n$ into the denominator of the summation term in the proof of the theorem in the previous section. Therefore, the upper bound given there can be multiplied by n to apply to coboundary OATs. This yields the following weaker result:

Theorem 4.1 *The expected height of an OAT from the coboundary distribution is less than*

$$en^3 \log n.$$

4.2 An $O(n^2)$ Bound on Expected Mean Pathlength

The height of a tree is the maximum pathlength of a node in the tree; the mean pathlength is the average taken over all nodes in the tree. The bounds that have been given are on the expected height rather than the expected mean pathlength of the tree, though it is the latter that corresponds to the expected performance of the algorithm. Since of course the former is always the larger of the two, it is possible to compute a slightly better bound.

Theorem 4.2 *The expected mean pathlength of an OAT under the boundary distribution or of a BAT under either the boundary or coboundary distributions is less than en^2.*

Proof: The expected values of $h_1, \ldots, h_i, \ldots, h_t$, where $t = \sum_{p=0}^{\lfloor\frac{n-1}{2}\rfloor} \binom{n}{p} = P_{\lfloor\frac{n-1}{2}\rfloor}$, are all less than $en^2/2$ because $en^2/2$ is greater than $e(n-1)\lfloor\frac{n-1}{2}\rfloor$. For all values of i greater than $P_{\lfloor\frac{n-1}{2}\rfloor}$, we must add to $en^2/2$ at least an additional amount

$$e(n-1)\binom{n}{\lfloor\frac{n+1}{2}\rfloor} \Big/ \binom{n}{\lfloor\frac{n+3}{2}\rfloor}.$$

In general, for any i which satisfies

$$i \geq 2^n - \sum_{p=j}^{n-1} \binom{n}{p}$$

the $E(h_i)$ gets an additional

$$e(n-1)\binom{n}{j} \Big/ \binom{n}{j+1}$$

for each $j = \lfloor\frac{n+1}{2}\rfloor, \ldots, n-1$. The total of all these contributions to the sum of the pathlengths is

$$e(n-1) \sum_{k=2}^{\lfloor\frac{n+1}{2}\rfloor} \frac{n+1-k}{k} P_k \leq e(n-1)n \sum_{k=2}^{\lfloor\frac{n+1}{2}\rfloor} 1/kP_k$$
$$< e(n-1)n \sum_{k=2}^{\lfloor\frac{n+1}{2}\rfloor} \binom{n}{k} \approx en^2 2^{n-1}.$$

So $2^n en^2/2 + en^2 2^{n-1} = 2^n en^2$ is a bound on the sum of the expected pathlengths, and therefore dividing by 2^n, the number of vertices, the expected mean pathlength is bounded by en^2. ▮

4.3 The Coboundary.

Recall that the boundary of a subset S of vertices of the n-cube is defined as the set of all vertices y not in S such that for some x in S, (x, y) is an edge of the cube. We define the *coboundary* of S to be the set of all distinct edges (x, y) where x is in S and y is not. The coboundary is clearly at least as large as the boundary. Now notice that in the proof of the dominance lemma for coboundary BATs, $B(i)$ could serve just as well as a symbol for a lower bound on the size of the coboundary as for the size of the boundary. Our next goal is to find a better value for $B(i)$, i.e., a lower bound on the size of the coboundary. This is provided via the following

Theorem 4.3 *(Harper, Bernstein) For any i, $0 < i \leq 2^n$, a subset of the vertices of size i with minimal coboundary size consists of the i smallest vertices, where the vertices are thought of as n-digit binary numbers.*

Proof. See (Clements, 1971, or Katona, 1974).

Theorem 4.4 *If i is greater than or equal to 2^d and less than 2^{d+1}, where d is less than $n-1$, then $2^d(n-d)$ is a lower bound on the coboundary of a set of cardinality i. If i is greater than 2^{n-1}, a bound can be obtained by replacing i with $2^n - i$.*

Proof. Let S_i be the number of interconnections (unordered pairs) within the minimal coboundary set of cardinality i. The coboundary has size $ni - 2S_i$. For $2^d < i < 2^{d+1}$ the members of the set are the same as for $0 < i < 2^d$ except for the "1" in the dth place. Therefore, $S_{i+1} - S_i$ equals $S_{i+1-2^d} - S_{i-2^d}$ plus the number of points with a "0" in the dth place that are adjacent to the $i+1$st point. This obviously equals one. Letting $R_i = S_{i+1} - S_i$ denote the ith difference, we have shown that

$$R_i = R_{i-2^d} + 1; \quad 2^d \leq i < 2^{d+1}.$$

This relation gives rise to the following as the R_i sequence:

0
1

1, 2
1, 2, 2, 3
1, 2, 2, 3, 2, 3, 3, 4
1, 2, 2, 3, 2, 3, 3, 4, 2, 3, 3, 4, 3, 4, 4, 5
. . .

The following proposition states some elementary properties of the R_i and S_i sequences.

Proposition.

1) $S_{2^d} = d2^{d-1}$

2) $S_{2^{d+1}} - S_{2^d} = (d+2)2^{d-1}$

3) $R_i = R_{i-2^{d-1}}; \quad 2^d \leq i < 2^d + 2^{d-1}$

4) $R_i = 1 + R_{i-2^{d-1}}; \quad 2^d + 2^{d-1} \leq i < 2^d + 2^{d-1} + 2^{d-2}$

We prove the theorem inductively, assuming it is true for i less than 2^d to show it is true for i less than 2^{d+1}. When $i = 2^d$, the set is a d-dimensional cube and the bound given by the theorem is tight. Letting $j = i - 2^d$, it is therefore necessary and sufficient to show that

$$nj \geq 2 \sum_{p=2^d}^{i} R_i$$

where $2^d \leq i < 2^{d+1} < 2^n$.

For the range $2^d < i \leq 2^d + 2^{d-1}$, by the proposition above, we have $R_i = R_{i-2^{d-1}}$ and the result follows from the inductive hypothesis.

For the range $2^d + 2^{d-1} < i \leq 2^{d+1}$, we note that it is sufficient to prove the result for the case where d equals $n-2$. In this case the symmetry between i and $2^n - i$ allows us to prove the result for the range $2^{d+1} \leq i < 2^{d+1} + 2^{d-1}$ instead. The bound given by the theorem obviously holds for $i = 2^{d+1}$. But R_i in this range equals R_{i-2^d}, and the bound has already been shown to hold for the range $2^d \leq i \leq 2^d + 2^{d-1}$. This completes the proof of the theorem 4.4. ▮

4.4 An $O(n \log n)$ bound for coboundary BATs.

We are now able to compute a better bound for the coboundary distribution. When d, the size of the partially constructed ordering, is ranging between 2^i and 2^{i+1} for $i \leq n-1$, the H_i values are growing at a rate bounded by a largest K process with $k_d = 2^i(n-i)/(n-1)$. This range for d consists of $2^{i+1} - 2^i = 2^i$ choices (iterations of the largest k process), so the increase in H_i values is bounded by

$$(2^i)e/k = e(n-1)/(n-i).$$

By the equivalence between d and $2^n - d$ mentioned in the theorem above, the increase in h values of the bounding process is mirrored as d ranges between 2^{n-1} and 2^n. Therefore, a bound on the last H_i value is given by twice the summation, from $i = 0$ to $n-1$, of $e(n-1)/(n-i)$,

$$2e(n-1)\sum_{i=0}^{n-1} 1/(n-i) \sim 2e(n-1)\log n.$$

Theorem 4.5 *The expected mean pathlength of coboundary BATs is less than $2en\log n$. The expected mean pathlength of coboundary OATs is less than $2en^2\log n$.*

Proof: The bound for coboundary BATs has been proved. To see how to adjust this bound to apply to coboundary OATs, we must examine the reasoning used in the Dominance Lemma. Again, suppose the random generation of a coboundary OAT is partially realized and that the ith vertex is about to be chosen. Fix j to be less than i and consider the jth vertex chosen. What is the probability that this vertex will be the father of the ith vertex? The jth vertex cannot have more than $n-1$ neighbors in the boundary. Each of these neighbors has at most a probability n/C of being chosen as the ith vertex, where C is the size of the coboundary. Therefore, the probability of being a father cannot be more than $n(n-1)/C$. This puts an extra factor of n into the computation of the upper bound for the case of coboundary OATs. ▮

CHAPTER 5.

APPLICATIONS TO PRINCIPAL PIVOTING AND THE SIMPLEX METHOD

5.1 LCP and BATs.

It is not essential to the concept of a BAT that there be a real valued function defined on the vertices. All that is necessary is that one vertex be optimal in some sense and that every other vertex have a father (a neighboring vertex) assigned to it in such a way that the resulting structure is a tree, i.e. has no cycles, or equivalently, traversing from vertex to father to father must eventually lead to the root. Consider the linear complementarity problem: given $q \in \Re^n$ and $M \in \Re^{n \times n}$, find $w, z \in \Re^n$ such that

$$\begin{aligned} -Mz + Iw &= q \\ z^T w &= 0 \\ z &\geq 0 \\ w &\geq 0. \end{aligned} \qquad (LCP)$$

The possible solutions to this problem involve choices of the complementary basis B, an n by n matrix whose ith column is the ith column of either $-M$ or the identity matrix, I. When M is a P-matrix, principal pivoting methods solve LCP by iteratively proceeding from a complementary basis to another that differs in the choice of one column. We let the ith digit of our binary n-vector equal zero if the ith column of B is taken from the identity matrix, and one if it is taken from $-M$. (This notation is unambiguous because the ith column of a matrix with positive principal minors cannot be the ith column of $-I$.) Then each choice of feasible basis corresponds to a vertex of the n-cube, and the pivoting algorithms proceed from one such vertex to an adjacent one. Thus the BAT model has features in common with principal pivoting methods for the linear complementarity problem. It is well known that such pivoting algorithms require $O(n)$ iterations in practice, though there are exponentially bad classes of instances (see Cottle, 1978). This accords exactly with the simulation predictions of expected BAT depth and with theoretical worst case BATs.

5.2 The Simplex Method

The Simplex Method solves the Linear Programming problem (LP) by passing from vertex to vertex on the polytope defined by the constraining hyperplanes. Suppose the number of variables to be k and the number of linear inequality constraints to be n, where $n \geq k$. Then there are at most $\binom{n}{k}$ basic solutions, each defined by the intersection of k hyperplanes. Each such point can be represented by an n digit binary vector with exactly k ones in it, to indicate which of the n hyperplanes define that particular point (or, equivalently, which variables are non-basic). If the point exists and is feasible, we assign it a function value equal to the objective function value; If some constraints are violated, we subtract the appropriate multiple of M, where M is some suitably large value, from the objective function value to get the function value. The Simplex Algorithm always proceeds by taking one column (hyperplane) out and putting one column into the basis set; so two of the binary vectors can be thought of as adjacent if they differ in two components. Thus, by restricting ourselves to just those vertices with k ones, and changing the definition of adjacency accordingly, we can apply the OAT and BAT models to the Simplex Method for linear programming. We call the space of n-tuples with k ones *n,k space,* and its elements, *n,k-vectors.* We say that two n, k-vectors are adjacent if they differ in exactly two components. In the following it is assumed that $k \leq n/2$ since the problem is the same for vectors with k ones as it is for vectors with $n - k$ ones.

Simulation results, as before, give a formula for expected tree depth that is approximately equal to the logarithm of the total number of points, $\binom{n}{k}$, in the space. For OATs with $k/n = \frac{1}{2}$, the average mean pathlength is roughly equal to $.5n - .2$, while for BATs it is approximately $.75n - 1$.

To get theoretical bounds, we need a lower bound on the size of the coboundary or boundary of a subset of the n, k-vectors. Unfortunately, no such bound is known. However, Kleitman and West have conjectured what the minimal coboundary subsets are:

Conjecture 5.1 (Kleitman, 1979) *How can we choose a k-hypergraph H on n vertices having d edges to minimize the coboundary? The conjecture is that there is an optimal H for any d so that either H or its complement as a k-hypergraph or the complements of the edges of H or the complements of these as an $(n-k)$-hypergraph do not contain any edges containing the n-th vertex.*

This somewhat confusing statement gives a complete answer to the problem if applied inductively. We clarify it and describe the break points that allow the inductive computation in the next section. Our goal, as before, is to derive numerical bounds on the size of the coboundary for different ranges of values of d. These bounds are stated below:

Lemma 5.10 *Let $Mcob(d)_{n,k}$ denote the size of the smallest possible coboundary of a subset of cardinality d in n, k space (according to the conjecture). Suppose $d \leq \frac{1}{2}\binom{n}{k}$ and $k \leq n/2$. Let p be the largest integer such that $k \leq \frac{1}{2}(n-p+1)$, so that $p = n - 2k + 1$. Then*

$$\binom{n-p-2}{k-2} \leq \binom{n-p}{k-1} \leq \cdots \leq \binom{n-1}{k-1} \leq \frac{1}{2}\binom{n}{k}.$$

Case 1: If $\binom{n-1}{k-1} \leq d \leq \frac{1}{2}\binom{n}{k}$ then $Mcob(d)_{n,k} \geq k\binom{n-1}{k}$.

Case 2:If $\binom{n-1}{k-1} \geq d \geq \binom{n-p}{k-1}$, then there is some i, $1 \leq i \leq p-1$, such that $\binom{n-i-1}{k-1} \leq d \leq \binom{n-i}{k-1}$, and

$$Mcob(d)_{n,k} \geq k\binom{n-i}{k} + k(i-1)\binom{n-i-1}{k-1}.$$

Case 3: If $\binom{n-p-2}{k-2} \leq d \leq \binom{n-p}{k-1}$, then

$$Mcob(d)_{n,k} \geq 2(kp + 2k - 2)\binom{n-p-2}{k-2}.$$

Case 4: Otherwise, there exists an integer m such that $2 \leq m \leq k-1$, and

$$\binom{2k-2m-1}{k-m-1} \leq d \leq \binom{2k-2m+1}{k-m}.$$

Then

$$Mcob(d)_{n,k} \geq (m-2)(2k-m+2) + (5k-5m+4+k(p-1))\binom{2k-2m-1}{k-m-1}.$$

We defer the proof of Lemma 5.10 until the next section because of its length. In the meanwhile we can now calculate an upper bound on the expected mean path-length of a BAT from the coboundary distribution in n, k space. Since $Mcob(d)_{n,k}$ equals $Mcob(d')_{n,k}$ where $d' = \binom{n}{k} - d$, we do the computation for the range $1 \leq d \leq \frac{1}{2}\binom{n}{k}$ and multiply the result by two. We also of course multiply by $ek(n-k)$ because each point has $k(n-k)$ neighbors. Case 1 of Lemma 5.10 yields

$$\frac{\frac{1}{2}\binom{n}{k} - \binom{n-1}{k-1}}{k\binom{n-1}{k}} = \frac{1}{2k}\left(\frac{n-2k}{n-k}\right).$$

In case 2 of Lemma 5.10, i ranges from 1 to $p-1$. The size of the range of d is $\binom{n-i}{k-1} - \binom{n-i-1}{k-2} = \binom{n-i-1}{k-2}$; $Mcob(d)_{n,k}$ is at least

$$k\binom{n-i}{k} + k(i-1)\binom{n-i-1}{k-1} = k\binom{n-i}{k}\left(1 + \frac{k(i-1)}{n-i}\right).$$

Therefore, case 2 yields

$$\sum_{i=1}^{p} \binom{n-i-1}{k-2} \Big/ k\binom{n-i}{k}\left(1 + \frac{k(i-1)}{n-i}\right).$$

The ith term in this sum equals

$$\begin{aligned}\frac{k-1}{(n-k-i+ik)(n-i-k+1)} &\leq (k-1)\left(\frac{1}{(ik)(n-k-i)}\right)\\ &= \frac{k-1}{k(n-k)}\left(\frac{1}{n-k-i} + \frac{1}{i}\right).\end{aligned}$$

The sum for case 2 is therefore less than

$$\frac{k-1}{k(n-k)} \log\left(\frac{(n-k-1)(p-1)}{k}\right).$$

Case 3 yields

$$\left[\binom{n-p}{k-1} - \binom{n-p-2}{k-2}\right] \Big/ 2(kp+2k-2)\binom{n-p-2}{k-2} \leq \frac{3}{k(n-2k+1)}.$$

Similarly, case 4 yields a summation with numerator

$$\binom{2k-2m+1}{k-m} - \binom{2k-2m-1}{k-m-1} = \binom{2k-2m}{k-m}\frac{3k-3m+1}{2(k-m+1)}.$$

If we discard the $(m-2)(2k-m+2)$ term that appears in the denominator, the summation term increases, so it is less than

$$\binom{2k-2m}{k-m}\left(\frac{3k-3m+1}{2(k-m+1)}\right) \Big/ (5k-5m+4+k(p-1))\binom{2k-2m-1}{k-m-1},$$

which turns out to be less than

$$\frac{3}{5(k-m)+4+k(p-1).}$$

Changing the index of summation and using the fact that $p-1 = n-2k$, case 4 yields

$$\sum_{i=1}^{k-2} \frac{3}{5i+4+k(n-2k)} \approx \frac{3}{5}\log(1+\frac{5k-10}{k(n-2k)}) \approx \frac{3}{n-2k}.$$

Let $r = k/n < \frac{1}{2}$ and let $n \to \infty$. Adding the contributions of the four cases and multiplying by $2ek(n-k) = 2er(1-r)n^2$, we get

Theorem 5.2 $2ern\log(n(1-r)(1-2r)/r) + 6enr(1-r)/(1-2r)$

is an $O(n\log n)$ bound on expected BAT height under the coboundary distribution.

Corollary. *Let $r = k/n < \frac{1}{2}$ be fixed, and suppose the minimal coboundary conjecture holds. Then under the coboundary distribution, the expected number of iterations required to solve LP by a simplex-type better adjacency algorithm, as $n \to \infty$, is less than*

$$2ern\log(n(1-r)(1-2r)/r) + 6enr(1-r)/(1-2r) \approx 2ek\log n.$$

5.3 Proof of Lemma 5.10.

We prove lemma 5.10 in several parts. Recall that $Mcob(d)_{n,k}$ is defined as the minimum coboundary size of a subset of cardinality d in n, k space. We define the *1-part* (respectively *0-part*) of a set of n, k-vectors to be the subset of vectors with nth coordinate equal to one (respectively zero). Thus $\binom{n-1}{k-1}$ is the size of the 1-part of n, k space. If $k \leq n/2$ and $d \leq \frac{1}{2}\binom{n}{k}$ then the conjecture says that if d is smaller than the 1-part of n, k space, the optimal hypergraph is in the 0-part of n, k space, otherwise it contains the entire 1-part of n, k space (Kleitman, 1979). Let $H(d)_{n,k}$ denote the optimal choice of n, k-vectors specified by the conjecture.

Lemma 5.3 *Suppose $k \leq n/2$; then $\binom{n-1}{k-1} \leq \frac{1}{2}\binom{n}{k}$. If $\binom{n-1}{k-1} \leq d \leq \frac{1}{2}\binom{n}{k}$ then*

$$Mcob(d)_{n,k} \geq Mcob(\binom{n-1}{k-1})_{n,k}.$$

Proof. If we let Δd equal $d - \binom{n-1}{k-1}$, then $H(d)_{n,k}$ includes the 1-part of n, k space and Δd elements from the 0-part. We compute the number of directed edges between these Δd elements and the 1-part: each member of the 0-part has k neighbors in the 1-part, so the number is $2k\Delta d$. When we add a set of Δd elements to the 1-part, we bring in $\Delta d(k)(n-k)$ new edges; what we must show is that the number of new interconnections between members of the set is not more than the number of new edges brought in, so that the new coboundary is not less than the coboundary of the 1-part. We claim that $k\Delta d(n-k-2)$ is greater than or equal to the number of interconnections (directed edges) within any set of Δd elements of the 0-part of n, k space. Obviously, a proof of this claim will prove lemma 5.3.

Proof of Claim: The number of interconnections is maximized when the coboundary is minimized. So the question is, what does $H(\Delta d)_{n-1,k}$ look like? We note that $\Delta d \leq \frac{1}{2}\binom{n}{k} - \binom{n-1}{k-1} = \frac{1}{2}[\binom{n-1}{k} - \binom{n-1}{k-1}]$, so $\Delta d \leq \frac{1}{2}\binom{n-1}{k}$, and so if $k \leq \frac{n-1}{2}$ we can use lemma 5.3 inductively. This is what we do in case (i).

Case (i): If $\Delta d \geq \binom{n-2}{k-1}$ and $k \leq \frac{n-1}{2}$, then inductively we know that

$$Mcob(\Delta d)_{n-1,k} \geq Mcob(\binom{n-2}{k-1})_{n-1,k}.$$

It is easy to see that the number of interconnections in $H((\binom{n-2}{k-1}))_{n-1,k}$ equals $\binom{n-2}{k-1}(k-1)(n-k-1)$. It is also evident that $Mcob(\alpha)_{n-1,k}$ + [the number of interconnections in $H(\alpha)_{n-1,k}$] $= |\alpha|k(n-k-1)$, which implies that

$$\begin{aligned}
&k\Delta d(n-k-1) - [\textit{number of interconnections in } H(\Delta d)_{n-1,k}]\\
&\qquad \geq k\binom{n-2}{k-1}(n-k-1) - [\textit{interconnections in } H(\binom{n-2}{k-1})_{n-1,k}]\\
&\Rightarrow [\textit{interconnections in } H(\Delta d)_{n-1,k}]\\
&\qquad \leq \binom{n-2}{k-1}(k-1)(n-k-1) + k(n-k-1)(\Delta d - \binom{n-2}{k-1})\\
&\qquad = (n-k-1)(k\Delta d - \binom{n-2}{k-1})\\
&\qquad = (n-k-2)k\Delta d + k\Delta d - (n-k-1)\binom{n-2}{k-1}.
\end{aligned}$$

To prove the claim in this case, we must show that the right hand side of the above is no more than $k\Delta d(n-k-2)$. Equivalently, we must show that $\binom{n-2}{k-1}(n-k-1) \geq k\Delta d$. Now

$$\Delta d \leq \frac{1}{2}\left[\binom{n-1}{k} - \binom{n-1}{k-1}\right] \leq \frac{1}{2}\binom{n-2}{k}$$

and

$$\binom{n-2}{k-1}\frac{n-k-1}{k} = \binom{n-2}{k},$$

so we are done with case (i).

Case (ii): Suppose $k \leq n/2$ and $\Delta d < \binom{n-2}{k-1}$. Then $k-1 \leq \frac{n-2}{2}$ so the conjecture says that $H(\Delta d)_{n-1,k}$ is in the 0-part of $n-1, k$ space. In this space, the total number of neighbors a point has is $k(n-k-2)$, so the total number of interconnections in $H(\Delta d)_{n-1,k}$ can't be more than $(n-k-2)k\Delta d$, which is the desired result.

Case (iii): The only case left occurs when $\Delta d > \binom{n-2}{k-1}$ and $k = n/2$. However, since $\Delta d \leq \frac{1}{2}[\binom{n-1}{k} - \binom{n-1}{k-1}]$ and $\binom{n-1}{k} = \binom{n-1}{k-1}$ when $k = n/2$, Δd must be ≤ 0 so this case is vacuous. Therefore, the claim and lemma 5.3 are proved. ∎

Remark.

$$Mcob(\binom{n-1}{k-1})_{n,k} = (n-k)\binom{n-1}{k-1} = k\binom{n-1}{k}.$$

Lemma 5.4

If $k \leq \frac{n-1}{2}$ *and*

$$\frac{1}{2}\binom{n-1}{k} < d \leq \binom{n-1}{k-1}$$

then

$$Mcob(d)_{n,k} \geq k\binom{n-1}{k}.$$

Proof. Let $d' = \binom{n-1}{k} - d$. Then $Mcob(d')_{n-1,k} = Mcob(d)_{n-1,k}$. Since $H(d)_{n,k}$ is in the 0-part of n, k space, $Mcob(d)_{n,k} = kd + Mcob(d)_{n-1,k}$ and so we need to know what $H(d')_{n-1,k}$ looks like. Note that $\binom{n-2}{k-1} \leq \frac{1}{2}\binom{n-1}{k}$, so either

$$\binom{n-2}{k-1} \leq d' \leq \frac{1}{2}\binom{n-1}{k}$$

or

$$d' < \binom{n-2}{k-1}.$$

In the first case, since $k \leq \frac{n-1}{2}$, Lemma 5.3 applies and

$$Mcob(d')_{n-1,k} \geq Mcob(\binom{n-2}{k-1})_{n-1,k} = k\binom{n-2}{k}.$$

Therefore,

$$Mcob(d)_{n,k} \geq kd + k\binom{n-2}{k} \geq \frac{k}{2}\binom{n-1}{k} + \binom{n-2}{k}.$$

So we need to show that $k\binom{n-2}{k} \geq \frac{1}{2}k\binom{n-1}{k}$, which turns out to be equivalent to $\frac{n-1}{2} \geq k$.

In the second case, $Mcob(d)_{n,k} = kd + Mcob(d')_{n-1,k}$ and, according to the conjecture, $H(d')_{n-1,k}$ is in the 0-part of $n-1, k$ space, so $Mcob(d')_{n-1,k} \geq kd'$. But then

$$Mcob(d)_{n,k} = kd + Mcob(d')_{n-1,k} \geq kd + kd' = k\binom{n-1}{k}$$

and Lemma 5.4 is proved. ▮

Lemma 5.5 *If* $k \leq \frac{n-1}{2}$ *and*

$$\binom{n-2}{k-1} \leq d \leq \frac{1}{2}\binom{n-1}{k}$$

then

$$Mcob(d)_{n,k} \geq k\binom{n-1}{k}.$$

Proof. By Lemma 5.3, $Mcob(d)_{n-1,k} \geq k\binom{n-2}{k}$. But since $d < \binom{n-1}{k-1}$, it follows that $Mcob(d)_{n,k} = kd + Mcob(d)_{n-1,k}$ and so the same argument for the first case of Lemma 5.4 works. ▮

Lemma 5.6.0 *Suppose*

$$\binom{n-i-1}{k-1} \leq d \leq \binom{n-i}{k-1} \leq \cdots \leq \binom{n-2}{k-1} \leq \binom{n-1}{k-1}$$

and i *is such that* $k \leq \frac{n-i}{2}$. *Then*

$$Mcob(d)_{n,k} \geq k\left[\binom{n-i}{k} + (i-1)\binom{n-i-1}{k-1}\right].$$

Proof. By Lemmas 5.4 and 5.5,

$$Mcob(d)_{n-i+1,k} \geq k\binom{n-i}{k}.$$

Since $d < \binom{n-i}{k-1}$, $H(d)_{n-i+1,k}$ is in the 0-part of $n-i+1, k$ space; in fact, $H(d)_{n,k}$ is in the 0-part of the 0-part of ... the 0-part of n, k space. Therefore,

$$Mcob(d)_{n,k} = Mcob(d)_{n-i+1,k} + kd(i-1).$$

Combining this with the above gives

$$Mcob(d)_{n,k} \geq k\binom{n-i}{k} + kd(i-1) \geq k\binom{n-i}{k} + k(i-1)\binom{n-i-1}{k-1},$$

which is the desired result. ∎

The reasoning used in the previous proof will be needed later; so we state it as a lemma.

Lemma 5.6.1. *Suppose $d \leq \binom{n-i}{k-1}$ where $k \leq \frac{n-i+1}{2}$ and suppose*

$$Mcob(d)_{n-i-1,k} \geq f$$

for some number f. Then

$$Mcob(d)_{n,k} \geq f + kd(i-1).$$

Proof. Since $k \leq \frac{n-i+1}{2}$ and $d \leq \binom{n-i}{k-1}$, we know that $H(d)_{n-i+1,k}$ is in the 0-part of $n-i+1, k$ space. Similarly, since $k \leq \frac{n-i+2}{2}$ and $d \leq \binom{n-i+1}{k-1}$, we know that $H(d)_{n-i+2,k}$ is in the 0-part of $n-i+2, k$ space, and so on – clearly the last $i-1$ coordinates of $H(d)_{n,k}$ equal zero. But any n, k-vector whose last $i-1$ coordinates equal zero has precisely k(i-1) neighbors whose last $i-1$ coordinates are not all zeroes. Therefore,

$$dk(i-1) + Mcob(d)_{n-i+1,k} = Mcob(d)_{n,k}$$

and the result follows immediately. ∎

Lemma 5.7.0. *Suppose $k = n/2$ and*

$$\binom{n-3}{k-2} \leq d \leq \binom{n-1}{k-1} - \binom{n-3}{k-2}.$$

Then

$$cob(d)_{n,k} \geq kd + (k-1)\binom{n-2}{k-1}.$$

Proof. Since

$$d \leq \binom{n-1}{k-1} = \frac{1}{2}\binom{n}{k}$$

it follows that $H(d)_{n,k}$ is in the 0-part of n, k space. Therefore,

$$Mcob(d)_{n,k} = Mcob(d)_{n-1,k} + kd.$$

Also, since $k + (k-1) = n-1$,

$$Mcob(d)_{n-1,k} = Mcob(d)_{n-1,k-1}.$$

Let $d' = \binom{n-1}{k-1} - d$. Then

$$Mcob(d)_{n-1,k-1} = Mcob(d')_{n-1,k-1}.$$

Since $k-1 \leq \frac{(n-1)-1}{2}$, Lemmas 5.4 and 5.5 apply to various ranges of d and d' giving

$$Mcob(d)_{n-1,k-1} = Mcob(d')_{n-1,k-1} \geq (k-1)\binom{n-2}{k-1}$$

for

$$\binom{n-3}{k-2} \leq d \leq \binom{n-1}{k-1} - \binom{n-3}{k-2}$$

$$\Rightarrow$$

$$Mcob(d)_{n,k} \geq (k-1)\binom{n-2}{k-1} + kd,$$

the desired result. ∎

Remark. *Under the conditions of Lemma 5.7.0,*

$$(k-1)\binom{n-2}{k-1} + kd \geq (k-1)\binom{n-2}{k-1} + \binom{n-3}{k-2} = (3k-2)\binom{n-3}{k-2}.$$

Lemma 5.7.1. *If $k = n/2$ and*

$$\binom{n-1}{k-1} - \binom{n-3}{k-2} < d \leq \binom{n-1}{k-1}$$

then

$$Mcob(d)_{n,k} \geq (k-1)\binom{n-1}{k-1} + d \geq k\binom{n-1}{k-1} - \binom{n-3}{k-2}.$$

Proof. Using terminology from lemma 5.7.0, $d' < \binom{n-2}{k-2}$, so $H(d')_{n-1,k-1}$ is in the 0-part of $n-1, k-1$ space. Therefore,

$$\begin{aligned} Mcob(d')_{n-1,k-1} &\geq d'(k-1) \\ &\Rightarrow \\ Mcob(d)_{n,k} &\geq d'(k-1) + dk = (k-1)\binom{n-1}{k-1} + d, \end{aligned}$$

the desired result. ∎

Remark. *Under the conditions of Lemma 5.7.1,*

$$k\binom{n-1}{k-1} - \binom{n-3}{k-2} \geq (3k-2)\binom{n-3}{k-2}.$$

Lemma 5.7.2. *If $k = n/2$ and $\binom{n-3}{k-2} \leq d \leq \binom{n-1}{k-1}$ then*

$$Mcob(d)_{n,k} \geq (3k-2)\binom{n-3}{k-2} = (\frac{3}{2}k-1)\binom{n-2}{k-1}.$$

Proof. This is just Lemmas 5.7.0, 5.7.1, and the Remarks. ∎

Combining Lemmas 5.7.2 and 5.6.1 gives

Lemma 5.8. *Suppose $d \leq \frac{1}{2}\binom{n}{k}$ and $k \leq n/2$. Let p be such that $k = \frac{n-p+1}{2}$. If*

$$\binom{n-p-2}{k-2} \leq d \leq \binom{n-p}{k-1}$$

then

$$\begin{aligned} Mcob(d)_{n,k} &\geq dk(p-1) + (3k-2)\binom{n-p-2}{k-2} \\ &\geq 2\binom{n-p-2}{k-2}(kp+2k-2). \end{aligned}$$

The only remaining case of Lemma 5.10 is $d < \binom{n-p-2}{k-2}$. If $Mcob(d)_{n-p+1,k} \geq f$ then $Mcob(d)_{n,k} \geq f + kd(p-1)$, so the problem is to find a bound on $Mcob(d)_{n-p+1,k}$ when $d < \binom{n-p-2}{k-2}$. Let us express this in different terms. We want a bound on $Mcob(d)_{2k,k}$ when $d < \binom{2k-3}{k-2} = \frac{1}{2}\binom{2k-2}{k-1}$. What does $H(d)_{2k,k}$ look like? Clearly it is in the 0-part of $2k, k$ space, so

$$Mcob(d)_{2k,k} = kd + Mcob(d)_{2k-1,k}.$$

Also,

$$Mcob(d)_{2k-1,k} = Mcob(d)_{2k-1,k-1}.$$

The problem now is to find a bound on $Mcob(d)_{2k-1,k-1}$ where $d < \binom{2k-3}{k-2}$. Again, $H(d)_{2k-1,k-1}$ is in the 0-part of $2k-1, k-1$ space, so

$$Mcob(d)_{2k-1,k-1} = (k-1)d + Mcob(d)_{2k-2,k-1}.$$

What is $Mcob(d)_{2k-2,k-1}$? Recall that

$$d < \binom{2k-2-1}{k-1-1}$$

and $k-1 = \frac{2k-2}{2}$, so applying Lemma 5.7.2, when

$$d \geq \binom{2k-5}{k-3},$$

we find that

$$Mcob(d)_{2k-2,k-1} \geq (3k-5)\binom{2k-5}{k-3}.$$

Now

$$Mcob(d)_{2k-2,k-1} + (k-1)d = Mcob(d)_{2k-1,k}$$

and

$$Mcob(d)_{2k-1,k} + kd = Mcob(d)_{2k,k}$$

hence

$$Mcob(d)_{2k-2,k-1} + (2k-1)d = Mcob(d)_{2k,k}.$$

We have shown that in the case $\binom{2k-5}{k-3} \leq d \leq \binom{2k-3}{k-2}$,

$$Mcob(d)_{2k,k} \geq (2k-1)d + (3k-5)\binom{2k-5}{k-3}.$$

In the other case, $d < \binom{2k-5}{k-3}$, let $k' = k-1$: we want a bound on $Mcob(d)_{2k',k'}$ when

$$d < \binom{2k'-3}{k'-2}.$$

Therefore, repeated use of the preceding will solve the problem. We have proved

Lemma 5.9.0 *Suppose $d < \binom{2k-3}{k-2}$. If $d \geq \binom{2k-5}{k-3}$ then*

$$Mcob(d)_{2k,k} \geq (2k-1)d + (3k-5)\binom{2k-5}{k-3}.$$

Otherwise,

$$Mcob(d)_{2k,k} = Mcob(d)_{2k-2,k-1} + (2k-1)d.$$

To get d in ranges, we repeatedly ap⌐ly Lemma 5.9.0, and add a $k(p-1)d$ term from Lemma 5.6.1, yielding:

Lemma 5.9.1. *Suppose $d < \binom{2k-3}{k-2}$. Then for some $m \geq 2$,*

$$\binom{2k-2m-1}{k-m-1} < d \leq \binom{2k-2m+1}{k-m}$$

and

$$\begin{aligned} Mcob(d)_{n,k} &\geq kd(p-1) + \sum_{i=1}^{m-2} 2(k-i) + 1 + (5k-5m+4)\binom{2(k-m)-1}{k-m-1} \\ &\geq (m-2)(2k-m+2) + (4k-5m+4+kp)\binom{2k-2m-1}{k-m-1}. \end{aligned}$$

This completes the last case of Lemma 5.10. ∎

CHAPTER 6.
PROBLEMS WITH MULTIPLE LOCAL OPTIMA

6.1 All Orderings Equally Likely

What happens when the problem is not local-global? For instance, what happens in general with LCP, or with the clique problem? In this case, a local improvement algorithm is not guaranteed to find a solution; recall that a forest of trees may be generated instead of a single tree. There are algorithms for some "hard" combinatorial problems, such as 0-1 integer programming or the travelling salesman problem, which make use of local improvement, and are justified because there is no known way to solve them exactly in a reasonable amount of time. Many artificial intelligence applications employ hill climbing, although the problems often turn out not to be local-global (Nilsson, 1981, Winston, 1977). The obvious questions to ask are, "what are the chances of a local improvement algorithm working?", and "how long will such an algorithm take?". These are equivalent to the questions, "how many trees are in the forest?", and "how high are the trees?".

Proposition 6.1 *Under the assumption that all orderings are equally likely the expected number of trees in the OAF is equal to* $(2^n)/(n+1)$.

Proof. Let x denote a vertex of the n-cube. For $x = 0$ to $2^n - 1$, let the random variable I_x equal one if x is a local optimum and zero otherwise. Then the expected number of local optima equals

$$E\left(\sum_{x=0}^{2^n-1} I_x\right) = \sum_{x=0}^{2^n-1} E(I_x). \tag{6.1.1}$$

The probability of x being a local optimum is the probability that it is the highest of $n+1$ vertices in the ordering (it and its n neighbors). If all orderings are equally likely, this probability is $1/(n+1)$. Thus

$$E(I_x) = 1/(n+1); \quad x = 0, \ldots, 2^n - 1. \tag{6.1.2}$$

Combining equations (6.1.1) with (6.1.2) yields the desired result. ∎

For problems with all orderings equally likely, then, a local improvement algorithm by itself has little chance of attaining a global optimum. This is true even for parallel processing versions that use multiple starting points (unless there are exponentially many). In view of the results for LG problems we might expect the average pathlength of nodes in the OAFs to be small. This is indeed the case:

Simulation Result 6.2 *Under the assumption, all orderings equally likely, the average mean pathlength is linear in n.*

Proposition 6.3 *Under the assumption, all orderings equally likely, the average mean pathlength is less than $\frac{1}{2}en^2$.*

Proof. This result follows easily from a computation similar to the ones in the proofs in the previous chapters.

6.2 Boundary Uniform Distributions

The assumption that all orderings are equally likely is appealing because it is easily stated. However, it may not be realistic. In particular, it fails to take into account correlations between function values of neighboring points. This is evidenced by the conflict between the exponentially many local optima in Proposition 6.1 and the high probability of there being one local optimum (as discussed in 2.3) when there is positive correlation. The boundary and coboundary distributions naturally incorporate some positive correlation between function values of adjacent points, but are of course not suitable for producing non-LG problems.

We now define two classes of distributions on non-LG problems which are extensions of the boundary and coboundary distributions. A probability distribution on orderings is said to be *boundary uniform* if all members of the boundary set of the first i vertices in the ordering have an equal probability of being the $i+1$st in the ordering. Similarly, a distribution is said to be *coboundary uniform* if the relative chances of boundary members are weighted according to the number of neighbors they have among the first i points in the ordering. There are no restrictions on vertices not in the boundary: their individual probabilities can differ widely, and the overall probability that a non-boundary member is chosen (and hence that another local optimum is introduced) can vary depending on what the first i vertices are. Since the pathlength of a starting vertex which is a local optimum is one, we can only decrease average mean pathlength by allowing additional local optima. Therefore, the bounds calculated earlier for the boundary (respectively coboundary) distribution extend to the class of boundary uniform (respectively coboundary uniform) distributions. We state this result in the following theorem.

Theorem 6.4 *The expected mean pathlength of an Optimal Adjacency Forest (OAF) or a Better Adjacency Forest (BAF), under any boundary uniform distribution, is less than en^2. The expected mean pathlength of a BAF from any coboundary uniform distribution is less than $2en \log n$. The expected mean pathlength of an OAF from any coboundary uniform distribution is less than $2en^2 \log n$.*

6.3 The Local-Global Property and NP-complete Problems

There appears to be a considerable difference between problems that are LG and those that are not. In particular, it seems that the well known NP-complete problems belong to the latter group. For instance, we have the following proposition:

Proposition 6.5 *In the travelling salesman problem, if two hamiltonian circuits differ only in the order in which two consecutive cities are visited, they are called adjacent. Then with this notion of adjacency, there exists a class of instances with exponentially many local optima that are not global optima.*

Proof. As the basis for our class of instances, we use a graph with six nodes labelled a, a', b, c, d, and e. The nodes a, b, c, and d form a rectangle with lengths $ab = cd = 24$, $ad = bc = 10$, and $ac = bd = 26$. Node e is located midway between the short sides and a little closer to side cd than to side ab, thus $ed = ec = 12.5$ and $ea = eb = 14.5$. The node a' is at some very small distance to node a, so its distances to other nodes are the same as for a. We remark that the circuit a, d, c, b, e, a', a is a local optimum but is not globally optimal since its cost is three more than the circuit a, d, e, c, b, a', a. The latter circuit is globally and of course locally optimal. Now construct n copies of this graph, setting all distances between nodes in different copies to 100, except that the a and a' nodes are at a distance of 20. Any circuit that starts at some a, goes around that copy with either of the two locally optimal circuits discussed above (leaving out a, a'), proceeds to another copy and goes around it with one of the two locally optimal circuits, etc., will be a local optimum in the n-copy graph. However, only one of these 2^n circuits will be globally optimal (the one that always used the second choice). We have constructed a graph with $6n$ nodes which has at least $2^n - 1$ local optima that are not global optima. ∎

We can further sharpen the apparent distinction between LG and NP-complete problems by showing that LG problems are essentially in $NP \bigcap co(NP)$ (and hence unlikely to be NP-complete). To be precise, we define the *set recognition version* of the optimization problem

$$\max_{x \in X} f(x)$$

to be the following question: Given an instance and a number k, does there exist an $x \in X$ such that $f(x)$ is at least k?

Proposition 6.6 *Suppose that, for some discrete optimization problem,*

$$\max_{x \in X} f(x)$$

there exists a notion of adjacency which assigns neighbors to each point in such a way that (1) the assignment of neighbors is independent of the instance of the problem (independent of the particular data), and (2) each vertex has polynomially many neighbors. Then if the problem is LG, its set recognition version is in $NP \bigcap co(NP)$.

Proof. If, given some particular data and number k, there is no $x \in X$ such that $f(x)$ is at least k, this fact can be proved in nondeterministic polynomial time by "guessing" the true optimum, showing its value is less than k, and verifying its optimality by comparing its value with the values of its (polynomially many) neighbors.

Theorem 6.7 *The clique problem is not LG under any data independent, polynomial assignment of adjacency. Also, under the ordinary notion of adjacency, (two subsets S and T of vertices are adjacent if one is a subset of the other and their cardinality differs by one), there exists a class of instances with exponentially many local optima that are not global optima.*

Proof. We play the adversary against an arbitrary fixed adjacency rule. The instance we construct will have n nodes, though we will not specify what size n is until later. Our target clique consists of the first $n/4$ nodes. It will be locally but not globally maximal. We connect all of these $n/4$ nodes with edges so they form a clique, and we do not make any more edges incident to these nodes. Consider the next $n/2$ nodes: there are $\binom{n/2}{n/4}$ subsets of order $n/4$ and $\binom{n/2}{1+n/4}$ subsets of order $(1 + n/4)$. By assumption, there exists a polynomial $p(n)$ which bounds the number of neighbors a subset can have. We choose n to be large enough that $np(n)$ is smaller than $\binom{n/2}{n/4}$. Then there must be a subset of the $n/2$ nodes with the properties that (i) it is of order $(1 + \frac{n}{4})$, (ii) it is not a neighbor of the target clique, and (iii) it contains no subset of order $n/4$ that is a neighbor of the target clique. We connect the nodes of this subset so as to make it a clique; all pairs of nodes not in the subset and not in the target clique remain unconnected. The subset is therefore the global maximum, but any neighbor of the target clique will not be a clique or will be of order less than $n/4$. Given an arbitrary polynomial adjacency rule, we have constructed an instance which violates the LG property. This proves the first statement of the theorem.

The ordinary notion of adjacency states that two subsets of the nodes of the graph are adjacent if one contains the other and their cardinality differs by one. We start with a graph whose nodes are labelled $1, 2, \ldots, i, \ldots, n$, and which is itself a clique. That is, for all distinct nodes i and j, the edge (i, j) is in the graph. Now we delete all edges of form (i,i+1). If a subset of the nodes contains i, it cannot contain either $i + 1$ or $i - 1$ and still be a clique. We can build up cliques $(v_1, \ldots, v_k, \ldots)$ by choosing v_1 equal to either node 1 or 2, and v_{k+1} equal to either $v_k + 2$ or $v_k + 3$. Since the subsets $(k, k + 1, k + 3)$, $(k, k + 1, k + 23)$, and $(k, k + 2, k + 3)$ do not have all of their edges, the cliques we build in this way are all locally optimal. There are more than $2^{n/3}$ of these because we made at least $n/3$ choices when constructing them. Moreover, most of them (all but $2n$, at least) are of order less than $n/2$, the maximal order achieved by the clique $(1, 3, 5, 7, \ldots)$. We therefore have exponentially many local optima that fail to be global optima. ▮

Note: Many known NP-complete problems, such as knapsack or three dimensional matching, originated in the form of optimization problems, so the idea of local optimality applies immediately. Some other problems, such as satisfiability, 3-colorability, or 2-partition, are ordinarily set recognition version (that is, yes/no) problems, so the concept of local optimality may not seem to apply. However, we have found that most such problems can be easily transformed into an optimization version. For example, the Boolean satisfiability problem becomes the problem of assigning Boolean values to the set of variables so as to maximize the total number of clauses that are true. The 3-colorability problem becomes the problem of assigning one of three colors to each node in the graph so as to minimize the number of pairs of nodes that have the same color and are connected by an edge. With 2-partition we try to minimize the difference between the sums of the two subsets; with subgraph isomorphism (given two graphs G and H, does H contain a subgraph isomorphic to G?) we try to find a mapping from the nodes of G into the nodes of H that minimizes the number of conflicts in the corresponding edge sets.

Using this notion of optimization versions of NP-complete problems, we can now discuss local and global optimality. We believe that all NP-complete problems have the property of exponentially many local optima. However, since this statement implies that $P \neq NP$, a proof will not be attempted. (If $P = NP$, then any LG problem in P such as linear programming would be NP-complete). We do remark, however, that many of the polynomial transformations used in NP-completeness results preserve the "topology" of adjacencies in such a way that local optima remain local optima. It is usually easy to show that some particular NP-complete problem is not LG.

The implications of the above results are that local improvement algorithms will work quickly but have little chance of finding the true optimum. Even with polynomially many different starting points, chances of success become small as n gets large. This assumes that the starting point or points are chosen at random; a heuristic that finds a good starting point or points, for instance, can of course make quite a difference. This is the case in algorithms for solving integer programming problems, where local improvement is an inexpensive way to improve a "good" solution (Hillier, 1969).

REFERENCES

Aldous, David, and Pittman, James (1980). Unpublished notes.

Clements, G. F. (1971). "Sets of lattice points which contain a maximal number of edges," *Proc. Amer. Math. Soc.*, **27**, 13–15.

Cook, S.A. (1971). "The complexity of theorem proving procedures,"*Proc. 3rd Annual ACM Symposium on Theory of Computing*, 151–158.

Cottle, R. W. (1978). "Observations on a class of nasty linear complementarity problems," *Technical Report 78-34*, Department of Operations Research, Stanford University.

Cutler, Leola, and Wolfe, Philip (1963). *Experiments in Linear Programming*, Report No. RM-3402, The Rand Corporation, Santa Monica, CA.

Dantzig, George (1963). *Linear Programming and Extensions*, Princeton University Press, New Jersey.

Dearing, P. M., Francis, R. L., and Lowe, T. J. (1976). "Convex Location Problems on Tree Networks", *Operations Research* **24**, 628–642.

Hillier, F. S. (1969). "Efficient Heuristic Procedures for Integer Linear Programming With an Interior," *Operations Research* **17**, 600-637.

Katona, G. H. (1974)."Extremal problems for hypergraphs," *Combinatorics*, M. Hall and J. H. vanLint, eds., Math. Centre Tracts 55, Amsterdam, 13–42.

Keller, J., (1980). "Random Drift on a Lattice," Departments of Mathematics and Mechanical Engineering, Stanford University.

Klee, V. (1970). *American Mathematics Monthly* **77**, 63–65).

Kleitman, D. J. (1974). "On an extremal property of antichains. The LYM property and some of its implications and applications," *Combinatorics*, M. Hall and J. H. vanLint, eds., Math. Centre Tracts 55, Amsterdam, 77–90.

Kleitman, D. J. (1979). "Hypergraphic extremal properties."

Knuth, Donald (1973). *The Art of Computer Programming*, Vol. I, Addison-Wesley, Reading, Massachusetts.

Nilsson, Nils (1980). *Principles of Artificial Intelligence*, Tioga Publishing Co., Palo Alto.

Reingold, Edward, Nievergelt, Jurg, and Deo, Narsingh, (1977). *Combinatorial Algorithms*, Prentice-Hall, Inc., New Jersey.

Ross, Sheldon (1981). "A Simple Heuristic Approach to Simplex Efficiency," U.C. Berkeley.

Wiener, H. (1980). "On A Random Point Motion Problem."

Winston, Patrick (1977). *Artificial Intelligence*, Addison-Wesley, Massachusetts.

DATE
FILMED
3-8